T — Y — N — E — & — W — E — A — R

TYNE & WEAR

ATARI 8-BIT USER GROUP

Newsletter of TWAUG

Software
Editorial
Buy & Sell
Hardware

TYNE&WEAR ATARI USER GROUP

Reviews
Help line
Section
Repair Info

Public Domain Library

ISSUE #13

JANUARY/FEBRUARY 1995

U — S — E — R — G — R — O — U — P

BRING YOUR EIGHT UP TO DATE with power products from COMPUTER SOFTWARE SERVICES

THE BLACK BOX

The BLACK BOX is an add-on board for the Atari 600XL,800XL and 130XE 8-bit computers. It is a T-shaped board that plugs into the PBI port of the XL computer, or the ECI and cartridge ports of the 130XE. Connectors for both types of computers are built into the BLACK BOX so no adapter boards are necessary. A cartridge port is available on the board itself for 130XE users.

The BLACK BOX provides many unique and useful functions. The four primary functions are:-

* RS-232 serial modem port
* Parallel printer port
* SASI/SCSI hard disk port
* Operating System enhancements

The BLACK BOX is $199.95 for the basic unit, and $249.95 with an onboard 64K printer buffer.
Shipping and Handling extra.

THE BLACK BOX ENHANCER

A must for all BLACK BOX owners. The BLACK BOX ENHANCER is a plug-in module for your BLACK BOX, enhancing the printer functions and adding an instantly available, full featured sector editor!

Installation of the BLACK BOX ENHANCER requires one simple solder connection. Only $49.95 plus shipping/handling.

THE FLOPPY BOARD

Our latest and greatest product. The FLOPPY BOARD is an add-on expansion board for the BLACK BOX interface. It allows the use of the same inexpensive floppy drive mechanisms used in IBM computers. The FLOPPY BOARD is the first floppy drive interface to support "high density" floppy drive mechanisms in either 5.25 inch or 3.5 inch. Built into the FLOPPY BOARD are our BLACK BOX ENHANCER and a version of our SUPER ARCHIVER to allow copying of protected disks for 3.5 inch format. Included with the FLOPPY BOARD is our program to read and write to IBM or ST formatted disks. This makes the FLOPPY BOARD the best way to transfer files to and from your 8-bit.

The FLOPPY BOARD is only $149.95 plus shipping & handling.

THE MULTIPLEXER

This device brings the power and flexibility of larger systems to your 8-bit. The Multiplexer is a collection of cartridge interface boards that allow up to 8 Ataris to read and write to the same drives (typically a hard disk), access the same printer(s), and talk to each other. It is the first practical networking system for the Atari 8-bit computer.

One "master" computer (any 8-bit) is equipped with the master Multiplexer interface. Then up to 8 "slave" computers hook up to this master, each having their own slave interface.

The "common" peripherals (things that are to be shared) are connected to the master. On each slave, all disk and printer I/O is routed through the master, so no extra disk drives are needed.

The Multiplexer sells for $199.95 for a master and two slave units with cable. Additional slave units are $89.95 each, plus shipping/handling.

THE SUPER ARCHIVER II

The SUPER ARCHIVER II edits and copies all enhanced density programs plus retains all the features of the SUPER ARCHIVER.

The SUPER ARCHIVER II is only $99.95 plus shipping & handling. NOTICE: if you already have THE SUPER ARCHIVER you may upgrade to S.A.II for only $29.95 plus shipping/handling. Software only.

THE BIT WRITER

The Super Archiver BIT WRITER is capable of duplicating even the "uncopyable" Electronic Arts and Synapse Syn-series, which employ 34 full sector tracks. The BIT WRITER must be used with the SUPER ARCHIVER

The BIT WRITER is only $79.95 plus shipping/handling.

THE ULTRA SPEED PLUS OS

The Operating System that should be in every XL/XE computer! The Ultra Speed Plus puts unbelievable speed and convenience at your fingertips.

Use any DOS to place Ultra Speed formats on your disks (with XF551 or modified 1050 drives), reading and writing at this speed with most programs. This high speed mode can be turned off for maximum compatibility.

Four simple solder connections are required for installation if your machine has a socketed OS ROM. The Ultra Speed OS is only $69.95 plus shipping/handling.

For more information on these and other 8-bit products:

CONTACT
COMPUTER SOFTWARE SERVICES
PO BOX 17660
ROCHESTER, NEW YORK 14617
USA

ORDERING LINE: (716) 429-5639
FAX: (716) 247-7158
BBS: (716) 247-7157

or contact T.W.A.U.G. we will do our best to help.

EDITORIAL

Who to blame!!!

John Matthewson
David Ewens
Max Gerum

The Editorial staff of TWAUG would like to wish a very happy and prosperous 1995 to all subscribers.

Good News for our readers

For the last few months, when I had a bit of time on my hands, I experimented with new fonts to find the best one for the newsletter. I know some of our readers had difficulties reading the reduced fonts in the issues. Starting with the next issue the print should be much larger and you will of course have more pages to read. It will not affect the quality or the price of the newsletter.

The contribution fees for home and abroad:

HOME	1 COPY	£2,00
-DO-	6 COPIES	£11.00
EUROPE	1 COPY	£2.20
--DO--	6 COPIES	£12.50
ELSEWHERE	1 COPY	£2.50
---DO----	6 COPIES	£14.00

REMINDER:

The book--The Complete and Essential Map--anyone unable to pay the full amount of the cash price can pay in instalments, whenever and whatever you can afford, at no extra charge.

The next issue will be ready by mid-March.

CONTENTS

EDITORIAL 3

BASIC TUTORIAL
by Ofer Saferman
and IMPROVING YOUR 1050
by Ron Fetzer 4

CODING CAPERS
by Andrew C. Thompson 8

PCXFORMER 2.0 and 3.0
8-bit emulator for ST & PC 9

INTERNET and YOU
by Jim Cutler 10

LETTER SECTION 11

BASIC COMPILER
by Andrew Mcintosh 13

CODES FOR LASERMAZE 17
SEARCHING FOR BOBBY FISHER
by Bill Hall
ULTRASONIC SWITCH
by Len Golding 18

CRACKING THE CODE
by Keith Mayhew 22

GAMES REVIEW
by Fred Meijer 28

GAMES REVIEW
by Mark Stinson 30

GAMES REVIEW
by Mark Fenwick 31

ANNOUNCEMENT
by TWAUG
DISK CONTENT 33

ADVERTS FOR R.GORE 34

MICRO DISCOUNT 35

ACPC
CURRENT NOTES & PHOENIX 36

BASIC TUTORIAL - ANIMATION 3

This is the third article on animation. In this article I am going to discuss scrolling. Since animation could be considered any type of movement, scrolling could definitely be considered as animation. Since we already know about character sets we will see how to combine scrolling with character sets to achieve animation.

What is scrolling? Well, scrolling is a technique that moves the contents of a screen up, down, left, or right. When the screen moves up or down, this is vertical scroll, and when the screen moves left or right this is horizontal scroll. Either the entire screen or selected lines can move.

Space games often use both horizontal and vertical srolling. The entire galaxy moves in the desired direction depending on the joystick. This enables us to have a larger playfield than the screen, thus adding more realism to the game.

There are two methods of scrolling. There is the course scroll which moves one byte at a time and appears jumpy, this is only useful for fast scrolling.

The other method is smooth scrolling which unfortunately could be implemented only in machine language. Because it has to be done in 'vertical blank interrupt' in order to prevent jumping. Don't worry, all the routines could be incorporated in a Basic program but it is a little tricky.

Remember character sets. Just imagine a colourful redefined character set that scrolls on the whole screen as a background for other moving objects.

Lets get to work. In order to perform any kind of scrolling we need to play with the display list (remember article 1?). If you remember, the fifth and sixth bytes of the display list hold the address of the begining of screen memory. Now, if we change these values we change where the screen memory begins. If we were to increment the numbers in a loop, and have a line written on the screen, we will see it scroll right to left and to the previous line and so on. But as I said before this is coarse scroll, its not very nice. The Atari computer has two scrolling registers. These scrolling registers cause a character on the screen to move pixel by pixel, either up down, or left to right. Now if we where to change these registers to almost scroll one character and then increment the fifth and sixth bytes in the display list, we will get a nice smooth scroll.

The problem is, if the scroll is smooth and the scroll register is being changed in Basic, the scroll will still occasionally jump and the result isn't nice. This is why we have to change the registers in machine language, during the vertical blank. Vertical blank is the time the electron beam has finished drawing the screen and gets back up and starts re-drawing it again. Changing the registers in this period avoids the ugly jumpiness.

Don't worry if you have no idea what I am talking about, I will supply the needed machine language routines that can be used in your program.

We still have to let the computer know which line we want to scroll, so we have to add to the mode instruction, of that line in the display list the value 16 for horizontal scroll, or 32 for vertical scroll. For example, if you are in graphics zero then this is Antic mode 2.

So just pick a line on the screen and check the display list instruction for that line, you will see that it is 2. So just add to the 2 either 16, or 32, or both, and poke it back to the display list. The computer will then know that you want to scroll that line. If you want to scroll at a higher speed, it can be done but it requires some machine language knowledge to change the routine. (The principle is very simple: instead of incrementing the scroll register by one you would increment it by 2 or 3 but not too much because you could loose smoothness).

This is very dry and technical stuff so check out the programs which I included and you will understand the basics better.

Don't miss the concluding part of this series because in my opinion it's the most interesting and important part - Player Missile Graphics.

For any questions, remarks, criticism, problems, etc. just write to me:

OFER SAFERMAN
21 BRANDE ST.
PETAH-TIQVA, 49600
ISRAEL.

IMPROVING YOUR 1050 DISK DRIVE.

by Ron Fetzer member of OL' HACKERS AUG

The inspiration and ideas for this article came from Ben Poehland, recently retired A.C. Managing Editor, and a member of OL' HACKERS.

AGEING?

Most 1050 disk drives by now are getting older and showing their age. Performance signs of age are occasional failure to format a disk, failure to read a good file, noise and sometimes inserting strange symbols into your text.

Before you make the upgrade you might want to check the condition of your power supply. Read the section DEVICES NEEDED and section VIII, VOLTAGE MEASUREMENT. This will give you a good idea of the condition of your power supply and if the upgrade is needed.

Once you have made the upgrade the 1050 will be just like new again, very reliable, quiet and will last you a very long time.

There are two types of 1050 disk drives. One is the TANDON and the other is the WORLD STORAGE. They are identical in most respects and are upgraded the same way.

IMPROVING THE 1050 DRIVE cont.

LABOUR OF LOVE

You must have skills in soldering and unsoldering. Budget about 4 hours for each drive upgrade. This is a labour of love and I would hesitate to impose on a friend with this project.

The description of the parts position is always with the disk drive front facing you. The Printed Circuit Board will be referred to in this article as the PCB.

DEVICES NEEDED

There are a few devices that you have to make yourself. You need TIE POINT EXTENDERS to read the voltages. You cannot reach the tie points otherwise easily. Take a 5 in. black piece of thin solid hook up wire. Remove 1 in insulation from one end and 1/4 in insulation from the other end. Take the end with the longer exposed wire and wrap it about 5 or 6 times around a sewing needle so it forms a little spring. You will now be able to slip this over the tie points on the PCB and be able to measure the voltages easily. Make two more exactly the same way with different color wire.

You will also need 4 SPLICERS. Take a very thin copper wire and wrap it about 10 times around a darning needle so it forms a spring about 1/4 long. Cut off the ends that are sticking out. Splicers are used to connect together wire components for soldering. They are also commercially available as COLMAN SPLICERS #1345-H

DISK DRIVE AGEING

What deteriorates in your disk drive is the internal power supply. The power supply is like the heart of a human being that pumps blood to the organs. The power supply pumps D.C. Voltages to all the I.C.'s and chips. This voltage has to be almost pure D.C. If there is a substantial A.C. component riding on top of the D.C. then your I.C.'s get confused and think they are receiving information pulses. A.C. or Alternating Current has a high and a low potential just like data pulses. That is the reason you have to upgrade your power supply to eliminate the A.C. ripple current on top of the D.C. voltage.

The solid state I.C.'s and other solid state devices will stay virtually like new if they receive the proper D.C. voltage and are protected from excessive heat.

While your disk drive is disassembled you should also clean all contacts, read-write head, belts, Pulleys and moving components etc.

At the end of this article I will give you a list of tools needed, supplies required and replacement components needed, and also where to get them.

FILTER CAPACITORS GET OLD

As they age, the filter capacitors in the power supply dry out and develop a high power factor. This results in letting A.C. pass through on top of the D.C. voltage. This voltage is usually called the ripple voltage.

Since this is a major overhaul of your 1050 disk drive we are going to replace them with premium filter capacitors which are larger than the original, high efficient, high temperature, long life types. We are also going to replace the 1N4001 1 Amp. Charge Pump Diodes with 3 Amp. heavy duty 1N5400 Silicon Rectifier Diodes. This will beef up the power supply so it that it is a new heavy duty long lasting supply.

THE JOB STEP BY STEP

I. DISASSEMBLY

1a) Turn your drive on its back and with a Phillips screw driver remove the 2 screws in the black front bezel. Then remove the 4 screws in the corners. Slide the top forward and out. Place it aside.

1b) Next we must disconnect all the jacks from the PCB so we can remove the mechanical motor assembly. The motor assembly rests on 4 pins with rubber cushions. Lift the front to see it. In the front on the right side of the PCB is a jack. Mark it and the PCB with White Out so that you know how to insert it again. Remove the jack. Do NOT pull on the wires but use the body of the jack to pull up on.

1c) In the back where the power supply is, there is a large jack on the left, and a smaller one on the right. They have to be removed. The left hand jack is really 5 jacks next to each other and they will come out individually. If you look next to the jacks on the PCB you will see the jack markings. On Tandon drives the body of the jack is also marked. On World Storage drives the jacks are corrugated and they are not marked. The easiest way to mark where the jack belongs is to take a piece of masking tape, write the jack number on it and glue it around the appropriate wires.

Here are the left handed jacks, front to back and the number of wires they have.

J11 has 4 wires
J12 has 2 wires
J1 has 4 wires(motor control)
J10 has 4 wires
J14 has 3 wires

On the right hand side there is a single jack J15. Mark it and the PCB with White Out so you know how to insert in later on. Remove it also.

1d) On World Storage drives there is one more wire, a ground wire, usually purple that is glued with apoxy to the heat sink. The easiest way to remove it is to cut it and later on to solder it again when you assemble the drive.

1e) You now can lift the motor assembly and remove it. Put it aside for now. You now have only the PCB and the bottom cover.

II PCB REMOVAL

You should wear a wrist grounding strap (see Tools) or have a grounded surface that you work on for the next steps.

IMPROVING 1050 DRIVES cont.

2a) If you remove the 4 rubber cushions on the motor pins it will be easier to remove the PCB. On Tandon drives about 2 inches from the front on either side of the PCB there are plastic snap holders that you must push back to remove the PCB. On World Storage drives you also have an additional two snaps in the back of the PCB. You also have 2 hold down screws on the PCB in the back and on on the left front of the PCB in some drives. You must remove them. The PCB will now come out quite easily. Put the bottom aside you will work only with the PCB.

III UPGRADING OF THE POWER SUPPLY

3a) As you look at the PCB in the back on the left side of the metal heat shield are two IC's mounted with screws and one on the right hand side. The rear left and right IC's are the voltage regulators. The left is a 7812 for 12 volts D.C.(Q8). The right is a 7805 for 5 volts D.C.(Q7). Both of these regulators should have a white paste or heat sink compound (see supplies) on their backs. If they seem to have very little compound on them remove the nut, bend the IC a little forward. Clean both surfaces with 91% Isopropyl alcohol. Put new heat sink paste on them and reapply the nut. This helps dissipate the heat from the IC. The front left IC usually has some kind of insulating material square clamped to its back. This is the TIP110 (Q6). Leave it ALONE!

3b) Next you are going to ground the metal case that covers most of the IC's. Cut a 5 in. piece of hook up wire. Strip the insulation from both sides. Find another nut that fits over the right side of the screw that sticks out from Q7 on the metal shield. Put the wire around the screw and put the second nut on top of it. Solder the other end of the wire to the side of the metal cage. This prevents RFI from your drive.

3c) Next we desolder the 3 big black filter capacitors C67, C68 and C71. On Tandon drives they have only 2 leads + and - . On World Storage drives they have 3 leads +,- and a dummy lead. You must desolder all of the leads to remove the capacitors. I used the SOLDAPULT desoldering tool from Radio Shack (see tools) and found it quite effective. After the capacitors are removed clean out the holes. To clean the holes put a little solder over it and suck it out again with the SOLDAPULT tool. The old capacitors were 4700 uf at 35 volts or less. The new ones we will install are going to be larger 6800 uf 35 volt, 105 degree C temperature, long life, high efficiency types.

3d) Remove the charge pump diodes CR15, CR16. They are in the middle between capacitor C71 and C68. They are clearly marked on the PCB. We are NOT going to desolder them, instead we are going to clip them near the body of the diode so we have two leads sticking up for each diode. The new diodes will get attached to these leads. The 3 Amp. diodes have thicker leads and would not fit into the holes on the PCB. Beside this is the easier way of doing it.

IV INSTALLATION OF NEW PARTS

4a) The first thing we are going to do is install the new 3 Amp. charge pump diodes 1N5400 in place of the CR15, CR16 1 Amp. diodes. Cut the lead of each new diode about 1/2 in. from its body and bend it 90 degrees. Take a solder splicer and fit it over each leg of the diode. Slip the other end of the solder splicer over the old leads of the diode sticking out of the PCB. The band of the diodes faces right. Solder the connections. Do this for both diodes. They both face to the right with the band.

4b) Next take your new filter capacitors, which are not wire lead types, but have snap legs, and see if you have to adjust the legs. They will not fit flush on the PCB but are about 1/8 in. above it. Adjust the legs so they fit into the holes of C67, C68 and C71 and solder them. The negative band faces to the right on all 3 filter capacitors.

V CLEANING

To clean the jacks contacts and I/O contacts use CREMOLIN R100 and B100 (See CN Oct. 90 page 24 "THE MAGIC JUICE") or if you do not have it use 91% alcohol (see supplies)

VI REASSEMBLY

6a) Put the PCB in the bottom cover and snap in the plastic retainers on the side of the PCB (4 on World Storage drives). If you had hold down screws for the PCB reattach them.

6b) Put the rubber cushion back on the 4 motor resting pins with the flat side up before putting the motor assembly back look at the belt on the bottom. Clean it with soap and water if it seems greasy. Clean the motor pulley and the flywheel with alcohol and reattach the belt.

6c) Take the motor assembly and put it on the 4 pins so they match the holes in the housing.

6d) Twist the wires around a few times and attach jack J15 on the right rear side. Twisting prevents RFI.

6e) Attach jack J6 on the front right side.

6f) Attach jacks J11, J12, J1, J10 and J14 to their pins on the left hand side. Twist J1 a few times around. This prevents RFI. J1 is the motor jack. It is sticking out a bit on Tandon drives.

6g) If you had cut the purple ground wire re-solder it and put insulation tape over it.

You are now finished with the major part of the reassembly of your drive. Plug the power transformer in to the drive, turn it on and see if everything is O.K. and the read/write head goes into the park position.

VII SPEED ADJUSTMENT

Use DOS 2.6f (black DOS) or any other disk drive speed checking program to see if the speed is correct. The 1050 is very stable in this regard and usually does not need to be adjusted. The correct speed is 288 RPM for an unmodified drive. The range can be from 280 to 296 without an adjustment.

IMPROVING 1050 DRIVE cont.

If you have to adjust the speed look for the potentiometer VR2. It is just to the right of jack J11. It is a small rectangular block with a tiny screw on top. Use a jewelers type of screw driver to adjust the speed to the right RPM.

VIII VOLTAGE MEASUREMENTS

Take the black TIE POINT EXTENDER LEAD that you made previously and slip it over TP15. TP15 is the ground. It is located in front of C68 and slightly to the left of it. It is clearly marked on the PCB. To the right of TP15 is TP13. Take the other extension and slip it over TP13. To the left of TP15 is TP14. Slip another extension over this tie point also. You now have 3 wires sticking out of your disk drive, 1 black and 2 others.

Take your Digital Volt Meter (DVM) or VOM and attach the ground probe to the black extension and attach the positive probe to the extension of TP13. Set the range switch to 10 volts D.C. Turn on the drive. It should read 4.9 - 5.1 v. D.C. Now attach the positive probe to the extension of TP14. It should read 12.01 - 12.03 v. D.C. These are not the most important voltages that you will measure. The most important voltage is the A.C. ripple voltage that is riding on top of the D.C. voltage.

To measure the A.C. ripple voltage get about a .1 uf 100v or higher mylar capacitor and with clip leads attach the capacitor to the extension of TP13 and the other end of the capacitor to your positive probe. Set your meter to its lowest A.C. range. The capacitor will block the D.C. and will let only the A.C. voltage pass. You should get a reading of .001v. A.C. or less. That is what I got. Do the same thing with TP14. That is 1 millivolt of A.C. RMS ripple which is excellent. I think it is even less than what I measured because my probe leads are not shielded. This voltage may be partly stray A.C. being picked up by the probe leads.

If you use an analog meter you do the same thing. On the A.C. ripple measurement the needle should not move or barely move at all.

The A.C. ripple is the quality measurement of your power supply. It should NOT have more than .002 v. A.C. RMS ripple riding on top of your D.C. voltage. The command signal to STOP READ on the Floppy Disk Controller Chip WD2793 on pin 27 is only -30 millivolts. If your power supply had an A.C. ripple voltage of only 11 millivolts RMS which translates into 30 millivolts Peak-to-Peak ripple A.C. it would be the same amplitude as the command signal and could easily confuse the chip. There are even smaller voltages in your disk drive. You can see why the power supply has to deliver clean D.C. voltages to your IC's.

IX LUBRICATING THE MECHANISM

9a) Use LUBE GEL (see supplies) and with a swab stick (see supplies) that is broken in the middle. Apply a small amount on the rails of the read - write head. Smooth it out with the cotton swab. DO NOT use Q Tips or oil in your drive. Q Tips are glued to the stick and when dunked into alcohol the glue then will spread over every surface you touch.

9b) Put Lube Gel on all the moving surfaces that eject the disk and are rubbing such as the tang that lowers the spindle to the disk etc.

9c) Take the cotton swab and with 91% Isopropyl Alcohol clean the read - write head.

9d) Rotate the pressure pad about 30 degrees with a screw driver to expose a new felt surface. The pressure pad is right over the read - write head and holds the disk to the head. 1/16 in. of felt should be showing below the plastic. If it is worn down to the plastic replace it. You can get a new one from BEST & CO.

9e) Clean the two I/O connectors in the back and the power supply jack with Cremolin or alcohol.

9f) To install the cover move it from the front to the back and install the 6 phillips head screws and tighten them. Your disk drive is now reassembled. Try it out and see if it works O.K.

X COSMETICS

On the back of your disk drive on top of the I/O connectors there is raised lettering. Mask the top and bottom with Scotch Tape and take a permanent type black felt tip marker and gently swipe over the lettering. Remove the Sotch Tape and the lettering will now stand out in black.

Mask the lettering on the front black bezel with Scotch Tape "OFF POWER ON". Also put tape in front and the back of the word "POWER". Take some White Out and put a little on a piece of plastic and with a smooth sponge pick up the wet White Out. Gently swipe it across the lettering. If you mess up you can remove it with alcohol. You now have the raised lettering in white on the front of your disk drive. Your disk drive is now in tip top shape. It is probably better than when it was new. It will work quietly and reliably for you for a very long time. In addition you have improved its appearance.

TOOLS REQUIRED

* A 30-40 watt pencil type soldering iron with a chisel tip.

* A desoldering tool like SOLDAPULT from Radio Shack #64-2120 $5.95. Use Lube Gel to re-lubricate it and establish vacuum.

* A long nose plier and a wire stripper and cutter.

* A phillips screw driver.

* A Static Drain Wrist Strap. Radio Shack #276-2397 $3.79

SUPPLIES NEEDED

* LUBE GEL from Radio Shack #64-2326 $2.79 (Do not use oil).

IMPROVING 1050 cont.

* 91% Isopropyl Alcohol from your drug store $1.00.

* Heat Sink Grease from Radio Shack #276-1372 $1.49.

* Long stem cotton swabs from Radio Shack (Do not use Q Tips).

* Thin hook up wire.

REPLACEMENT PARTS

* For each disk drive you need 3 filter capacitors. They are not available from Radio Shack. Order them from:

DIGI-KEY CORP.
701 Brooks Avenue South
Thief River Falls, MN 56701-0677
Tel:1-800-344-4539
Fax:218-681-3380

They are 6800 uf 35 volt capacitors. Digi-key #P6465ND each costs $4.04.

*You will also need two 3 Amp. diodes for each drive. They are 1N5400 Silicon Rectifier Diodes. Digi-key #1N5400GI.

The diodes are also available from Radio Shack #276-1141. They are two for $.99

If you have done the upgrade you can now pat yourself on the back for a job well done. Your disk drive is now as good as you can make it. It will last a long time and you will now have the peace of mind knowing that you have done all that is possible to do.

This article first appeared in 'The OL' HACKERS' ATARI USER GROUP INC. Newsletter November/December 1994 issue. Reprinted by TWAUG with permission.

CODING CAPERS.

by Andrew C. Thompson

And we return for another part of the programming column, this issue being something completely different, I'm going to take you on a voyage of complete and utter insanity and a small break from too much programming. Yes, you guessed it... I'm going to play the piano totally out of tune until your brain (for those who still have one) melts out from your ear lugs! There's a slight snag, however, we are not going to program musix, instead I have treated you to some completely original works that have been lying dorment in my cupboards for about a year now.

These pieces if you like (10 in all) all have the filename extension .SNG and you'll need to load them into the Torsten Karwoth SoundMonitor Professional to listen to them. You should find that musix package on this issues disk unless Dave has decided to leave it out. Let me see now, the pieces are:

CODING CAPER cont.

SPRING, SYZZLE, ROARX, YABUTE, BEATZ,
FRIZ, SYZZLE2, CRYSVAR, LIQDEM
and the last is named TING.

SPRING is a simple song using simple chords written of course, by a simple person. Chords I hear you say, but what are they. Well, Chords are a group of notes banged on the piano or plucked on the guitar simultaneously. I.E, a piano demonstration: Rightyho Elgar; lift right hand, spread selected fingers and hold tips of fingers at same height... quickly thrust down and baang#! Yes! A CHORD was played. It might be out of tune as far as we know but at least that's how a chord is played.
Later in this song there comes some so called Arpeggios (someone told me how to spell this word!), this is where each note of a chord is played in quick succession. So this example on the guitar would probably sound like: frankly Hank, Ding Ding Ding!

The rest of these short tunes have their own little peculiarities, such like the SYZZLE ones, these sound very strange, like a few pieces of bacon in the frying pan being fried at different speeds. Now there's a high-tech tip to achieving different frequencies in the volume-only bit mode for all you intermediate programmers. ROARX is also a distant runoff of an arpeggio style movement, but each group of notes are from 3 octaves, this way a strange bassline of whitenoise has introduced itself. Whitenoise being something like the noise a pressure cooker makes when the spuds are just about done.

FRIZ is something completely different originally designed at a slower speed but played fast, introduces violin and vocal sound affects, listen to it and see what I mean. Now, TING and CRYSVAR are both short variations of the original tune CRYSTAL which came out with the Soundmonitor package. Anyhow, YABUTE is here for you to laugh your cotton socks off since it's my strange attempt to medievalism. The last tune, LIQDEM or LIQUID DEMISE appeared on my EGO demo not so long back, aha! you can see how it's made and then sling it where it belongs!!

Anyhow, leaving the subject of earache, let's move into some headaches and give some quick answers to some questions that have come my way:

Q: How does one achieve moveable shapes greater than 8 bits in width?

A: There are several ways; the 1st is to use more than 1 PMG for the same shape, a 2nd way is through software sprites (self created and controlled), whilst a 3rd way would be to put your image in screen memory and scroll the memory. There are always limitations, but depending on the application use the most suitable method.

Q: How would I give weight to an object, meaning to slow something down to a halt and not to just stop it like as though it has crashed?

A: One way would be to keep a speed variable (SV) and a table, ie:

CODING CAPER cont.

SV:	TABLE
0	0
1	0.25
2	0.5
3	0.75
4	1
5	1.25
6	1.5
7	1.75
8	2

If a racing car is travelling with the speed variable on maximum 8, then the scrolling or pixel increment would be 2. If you push left on the joystick then you decrease SV and thus need to read a different pixel increment. This of course is just one method for one purpose, it's all to do with experimentaion.

Q: How did I achieve the Graphics 0 text sine-wave in my demo on TWAUG's 1st issue disk?

A: Very quickly, hold vertical positions for the characters and POKE them in row by row, character for character. Now, my technique was to give each character a direction, 1=down, 0=up. With a small routine, check the direction flags for each character and adjust their vertical positions as such, if any vertical positions exceed maximum or minimum allowances then change direction flag, then just re-loop, which re-POKEs them and starts all over.

There were other small routines to control the timing of luminance changes and text address changes whilst the lines between the text are just Blank scan-lines in the display memory. Ofcourse, don't forget that the vertical positions have to be multiplied by 40 to maintain the correct rows.

Q: Why can't someone detail DLI's in a somewhat simple manor?

A: Yeah! I agree, the person who posed this question to Dave apparently already has the Complete and Essential MAP and in it is a full tutorial on DLI's which I consider very direct. Anyone looking to utilize the power of DLI's should read this appendix, and if you don't have the book then so be it on your head.

Well then folks, that about concludes another, different, but partially helping (and partially hindering) Coding Capers Column.

PC XFORMER 2.0.

First published in Current Notes Aug/Sep 1994 issue, reprinted by TWAUG with permission.

Six years after releasing the Atari ST version of the 8-bit emulator to Compuserve and GEnie, we released PC Xformer 2.0 the same way--free of charge and available on Compuserve, GEnie, the Internet, and most Atari BBSs as well as in the CN library (PC 1).

PC Xformer 2.0 is a 16-bit implementation of an Atari 400/800 emulator.

XFORMER 2.0. cont.

That means that it can run on any 16-bit Intel compatible PC, include those using the 286, 386, 486, and Pentium chips, as well as PCs that can emulate the 286.

PC Xformer 2.0 is the first and only Atari 8-bit emulator for either the ST or PC that support all these features.

* 48K RAM with built-in Atari 400/800 operating system and Atari BASIC

* Display all ANTIC text and graphics modes, including GTIA modes

* Display player missile graphics (no collision detection)

* Handles display list interrupts, IRQ, and NMI interrupts

* 256 colour GTIA palette

* Reads and writes to ST Xformer and SIO2PC generated disk images

* Reads MS-DOS files directly

* Switches Atari BASIC on/off with one keystroke

* Joystick emulation using the keyboard

* Slow and fast modes of emulation for faster PCs

The speed of Atari 800 emulation is proportional to the speed of your PC, with a 33 Mhz 386 approximately giving the normal speed of an Atari 800. Twice the speed of an Atari 800 when running on a 486/33, and faster yet on faster PCs.

486 and Pentium users can use the slow/fast option to slow down the emulator to normal speed to make games playable, or speed it up to allow graphics and mathematical programs to run as fast as possible.

XFORMER 3.0.

PC Xformer 3.0 is a 32-bit implementation of an Atari 800XL/130XE emulator. It does not run on the 286 processor, and thus requires a 386 or faster. By using 32-bit features, of the 386 chip (the same reason why Windows NT and OS/2 v2.1 also do not run on a 286), PC Xformer 3.0 adds these additional features, which truly make it a complete 130XE replacement:

* 800XL/130XE memory bank switching
* Player missile graphics collision detection
* Sound card, joystick port, modem port, and printer port support.

PC Xformer 3.0 runs on any 386, 486, Pentium-based PC with at least 640K of RAM and a VGA compatible display, so only a small number of PC users will be unable to run PC Xformer 3.0. A sound blaster card and game card are optional.

Users of PC Xformer 2.0 can upgrade to version 3.0 for only $29.95. You can order directly from Branch Always Software.

XFORMER 2.0 cont.

For additional information or to place an order write to: Branch Always Software, 14150 NE. 20th Street, Suite 302, Bellevue, WA 98007, U.S.A. Or fax to: 206-885-5893.

NOTE: If you are interested in obtaining PC Xformer 2.0, you can purchase it from the TWAUG PD library.

The TWAUG package contains three disks, one 5.25 in Atari DOS format, and two 3.5 PC format disks. The Atari disk contains full documentation for building your own IO2PC interface, also on the disk are all the utilities you will need. The first 3.5 disk contains all the PC utilities and documentation for running the IO2PC interface. The second 3.5 disk contains the PC Xformer with full documentation.

The PC Xformer package is available for only £4 which includes postage and packing. Buy this PD version, and if you like it, you can place your order for the upgraded version from Branch Always Software at the above address.

ATARI 8-BIT,THE INTERNET AND YOU

With all the media hype about the "Super-highway" and the Internet it may interest you to learn that the humble 8-bit will give you access to most of it's many 'goodies'.

THINGS YOU MUST HAVE

1.An 8 bit with a diskdrive
2.A modem
3.A comms programme - I use the excellent BOBTERM.
4.A phone connection for the modem.
5.An Internet provider - Compuserve,Demon,Cix
6.A Credit card - you gotta pay mate!!!

WARNING

The provider charges you to join and for the time you use. They offer different things. I use CIX (Pronounced KICKS) its simple, very friendly, costs more than Demon,less than Compuserve. If you want to try out CIX use your modem to dial up 081-390-1244 and follow the prompts - they offer a free tour. You also pay the connection to the UK end.

THE NET IS ADDICTIVE - WATCH THE PHONE BILL

Cix offers E-MAIL, conferencing (Even an Atari 8bit one) Chat and Internet.

TELNET JARGON

TELNET - A means of getting directly into a distant computer or board.

EXAMPLE:

I telnet to The Well a board in Sanfrancisco. I instruct CIX to 'go internet' and at the 'ip' prompt type TELNET WELL.COM. Once Telnet connects me I use their board. The connection UK/Sanfrancisco is at UK local rates. There are lots of 'sites' taking TELNET

INTERNET cont.

FTP - File Transfer Protocol allows entry to a distant computer and the download of text files from that end. I emphasize TEXT as the 8 bit's memory limits frustrate Graphical or Zip files.

There is a convention of courtesy in FTP. The host is allowing you the courtesy of using it often, this very very expensive facilities.

GOPHER: A super quick means of locating specific files for FTP'ing.

ARCHIE: Basically a similar thing,doing the job differently.

WWW: The latest 'toy' The BBC board is a Web one. You recognise www sites by the prefix \HTTP\, I haven't yet tried it but BOBTERM set to VT100 emulation should allow Text access. If you're into Esoteric things, H.M.Treasury has just gone on WWW!! Gee Wow.

IRC- International Relay Chat- talk to other users on a direct on screen mode...Similar to a phone but your conversation's on one half, the guy or gal you talk to is on the other half of the screen.

I found the guys on CIX very helpful to a novice NET SURFER!!

NO!!! I don't have shares in CIX - I used COMPUSERVE at the beginning, it's good, has lots of goodies to download once you fathom where they are, but it's V*E*R*Y American and costly if you only have 300/1200 baud on your Modem.

Whatever service you choose you'll get E-Mail. For those who haven't ever used E-mail the speed would be a surprise. Your 'mail' goes as soon as you've sent it, the reply sits in your 'in basket' for the next time you look for it. SNAIL MAIL is the one you pay 19p or 25p for a quicker snail!!!!

If there's anybody who needs more info, drop a line to TWAUG - my limited (very) experiences are at your disposal.

Jim Cutler
E-mail : jcutlerfcix.compulink.co.uk
: amarofwell.com.sf.ca.us

Bear in mind that should you wish to connect to a Bulletin board, say in Holland or Germany, you pay the international phone charges. If it has a TELNET or FTP facility you're paying at UK charges - USE Cheap weekend phone costs.

LETTER SECTION

Mr Andrew MCINTOSH
21 ROMAN ROAD
BONNYBRIDGE
STIRLINGSHIRE
FK4 2DE
SCOTLAND

Dear TWAUG,

For Mailbag: please include my full international contact address with this letter, as it allows persons from outside the UK to write direct to me!

I would like to thank the following, Allan Hitchen, Andrew C. Thompson, Brad Rogers, Dennis Foggerty, Mark Watson, Raphael J. Espino, Richard Gale, Robert Mcgrath, Peter Panayiotopoulos, and John Picken for info that they have sent either to TWAUG, NAU or myself direct.

In the Mailbag issue 11, Allan Hitchen referred to the RF modulator being faulty if the machine was made in Hong Kong, actually my machine is made in Taiwan, but there's no way of knowing if the modulator was made in Taiwan too?

Mark Watson wrote suggesting that the grainy effect could be a fault in either PIA or GTIA IC's or in both chips.

Actually a friends 130XE also suffers from this vertical grainy effect. Around 6 years ago he also has a scart lead going to a CBM 1084S monitor and as far as I can remember the grainy effect was still present. After bypassing the RF modulator altogether, which would appear that the RF modulator was not at fault after all. But since it was a long time ago, I cannot be sure for certain, and since I have not seen any other 130XE or other upgraded XL's. I have been wondering if this grainy effect is a side-effect of the extended memory?

Below is an example of what the grainy effect my 256K XL is producing. My key coding is as follows, ' ! = a light colour' and ' | = a dark colour', in the width of one 8*8 bit character. Each vertical grain can be split into a width of 2 bits and in the space of one 8*8 bits of character there is "!|!|" or a light, dark, light, dark vertical bands throughout the entire screen, regardless which graphics mode is on screen!

!!!

LETTER SECTION continued

Finally can anyone tell me if Optimized Systems Software Incorporated still exists and can they supply the contact address. I would like to get in touch with Mr. Bill Wilkinson, the Copyright holder of Atari Basic, DOS 2.5, Assembler Editor and very much more besides, or does anyone know Mr. Bill Wilkinson's contact address?

Yours faithfully,

Andrew MCINTOSH (ASI Industries.)
PS: 'ASI Industries.' is my computer-hack-name not my company'

Hello again,

Please find enclosed my subscription renewal. I enjoy using the Atari 8-bit, (used to compose this very letter!).

I've been meaning to put something down on paper for a while now, but a thousand other things always appear before my eyes. The decorating, the leaking plumbing, you name it!

I also subscribe to other 8-bit info-gatherers and am pleased to note that you don't nag over lack of support, but the opposite. 'Think positive', I remember it has frequently been drummed into my ears as a student.

I am also pleased to add that the spelling has vastly improved, (is it thanks to the local night school? Or some spell-checker, which one?) I am not that good at spelling either, but when you're learning you notice all other poor spellers.

I use an old daisy-wheel printer with my 8-bit. You would think that I could easily up-grade to a fancy dot-matrix, but then old printers like mine can be picked up very cheaply. They don't draw fancy pictures, but work very well with word-processors. There's another advantage. The print never fades, (I've noticed that the work I produced only last year on a dot-matrix has shown to be 'yellowing'). I'll guess others will chant on over the disadvantages of the daisy-wheel, but the cost of its purchase (2nd-hand) outweighs ALL arguments!

My young children have now taken priority over the 8-bit. (I get a look-in when they climb into bed). There's a vast number of games on the machine, either cartridge, tape or disk. They seem to like the speed of loading with the Roms, though don't complain when I help loading via the disk. I guess that's something to do with my eldest, who once posted a piece of paper into the drive. It took months to figure out the problem and it still fails on the odd occassion. I must learn not to blame him as 5 years have gone by since.

I overheard a student boast that he managed to obtain a word-processor for ninety pounds. I interrupted to inquire about the printer, (to gloat). He told me, there was NO printer with it!

I didn't dare tell him that my 8-bit with printer costs only 39 pounds.
Keep up the good work.

On request, the name of this letter writer has been withheld.

REPLY

I am very pleased to hear that you can see an improvement in my spelling. I am doing my very best. But it really isn't my spelling that needs improving, I am fairly good at spelling and if I am in doubt about a word I consult the dictionary which is always by my side. I also use a very good spell checker, but the checker doesn't tell me when a word is missing in a sentence or when placed wrongly, nor does it tell me when there's no capital letters after a full stop. You cannot beat your eyes for proof reading, but it doesn't help when your eyes are getting old and tired, I do overlook some spelling and typing mistakes. But what's wrong with one or two misspelled words among friends, even the professionals make mistakes occasionally.

Max.

Hello David, John & Max.

Minor additional info to that nice book you published C.& E.M. by A.T.. In part 2 of the book on page 298, regarding free bytes location 653 (26D) is fixed to 252 (dec) and is unalterable. (My 8-bit would suck if it's the only one that has this). A few other locations free for use not mentioned are as follows:

651 (28B)-vacant.

693 (2B5)-vacant, screen line (default 255), read book for explanation on location.

1017 (3F9)-vacant.

Additional info for free bytes, as follows:

47 (2F)-used in Basic for PUT operation, read book for explanation on location and similar locations.

1021-1023 (3FD-3FF)- not used, read book for explanation on locations during initial cassette/disk loading operation, (see CASBUF).

Just a few brief notes, I hope they make sense to all interested.

Thank You

Amish Aziz

BASIC COMPILERS FOR THE ATARI

A COMPARATIVE VIEW.

Reprinted from Analog computing issue 11; Apr/May/83.

ABC, 48k disk, by Monarch Data Systems.

The Basic Compiler, 48k disk, by Datasoft.

BASM, 32k disk, by Computer Alliance.

Originally written by Brian Moriarty.

The world is full of Atari Basic programmers listing for speed. They squirm with envy as the desciples of C, Forth expound the virtues of those fast and exotic languages. They gaze with wonder upon Assembly language programmers who wield their mysterious powers at 1.79 mhz.

Why this insatiable craving for faster programs? The answer is simple: Games. Every serious Atari user has the secret desire to create the ultimate computer game. A dazzling tour-de-force that would make Tempest look like Pong. Unfortunately many would be Chris Crawfords don't have the time to master more than one programming language and there's no need to guess which excruciatingly slow language that one usually is?

If you've ever been frustrated by the speed of Atari Basic, then a Basic compiler may just be what you need. The recent releases of three new compilers for the Atari offers Atari programmers a long overdue alternative to Basic, that the Radio-Shack (Tandy) TRS-80 and Apple II users have been enjoying for years.

WHAT IS A COMPILER?:

A compiler is a utility program that reads a program written in Basic and translates it into a lower-level code, that executes faster than the original. A compiled Basic program is completely self-contained. It is treated exactly like a binary DOS object file and does not need the Basic rom or any other special software to run.

Monarch Data Systems' ABC (A Basic Compiler), Datasoft's Basic compiler and Computer Alliance's BASM are significantly different in terms of features, performance and cost. Since ABC was on the market ahead of the others, we'll examine it first.

MONARCH DATA SYSTEMS ABC COMPILER:

ABC is a single pass integer compiler, 'single' pass means that your Basic program is only scanned once as it is being compiled. 'Integer' means that numbers are stored in straight 3-byte binary instead of the usual 6-byte floating point format used by Atari Basic. The elimination of the floating point maths is one of the main reason for the speed of ABC.

The best way to understand ABC is to review what happens when you compose a Basic program. Each time you press the (RETURN) key over a line of Basic code, the instructions are tokenized into a special internal code that can be understood by Atari Basic.

ABC takes this process a step further. It reads the tokenized file produced by Atari Basic and translates it into a even more compact form called 'Psuedo-code' or P-code. This P-code is then linked to a small machine language program called a 'Run-time interpreter', which reads and executes each P-coded instruction.

The big difference between tokenized Atari Basic and ABC's P-code is it's conciseness, by using only whole number integer arithmetic and a more efficent memory management scheme. ABC simplifies the execution of each command in Atari Basic repertoire, the result is a significant increase in the speed of the compiled program. According to Monarch, the speed improvement factor can range between four and twelve times, seven times being a reasonable average.

It should be noted that the P-code is not directly straight 6502 assembly language. It is essentially a series of pointers into the Run-time interpreter, much like a Forth program. You cannot LIST, Dis-assemble or make any significant sense out of the P-code without a detailed understanding of the ABC's Run-time interpreter. This is an important feature if you are planning distributing your compiled software, because the code will be protected against all but the most determined pirates.

EASY TO USE:

Experienced Atari Basic programmers should have no trouble using ABC. First you Save your completed Atari Basic program to disk, then you remove all cartridges from your Atari and boot the ABC disk. Once ABC is loaded it asks for the name of your Atari Basic source file and the name of your target file, the target file will become the compiled version of your program.

ABC next writes a copy of the run-time interpreter out to the target file. It then scans your Atari Basic program and translates it into P-code, one line at a time. Finally, the P-code is appended to the Run-time interpreter, and you are left with a binary format disk file. This file can be loaded and executed though Atari DOS using option 'L', the original Atari Basic source program file is completely unaffected.

A couple of different Run-time libraries are included on the ABC disk. These provide a choice of loading addresses to match different memory configurations and DOS requirements. There is also a clever little program called 'MKRELO' that makes your compiled program re-locatable, a handy feature for commercial development. It assures that your software will run on virtually any Atari computer with enough memory.

THE DATASOFT COMPILER:

Datasoft's Basic compiler is a four pass utility, that converts Atari Basic programs directly into 6502 machine language. Because machine language does not need to be interpreted, the execution speed of the compiled program can be very impressive.

BASIC COMPILER continued

Datasoft claims a speed improvement of 5 to 20 times over an original Atari Basic program.

Like Monarch's ABC a Run-time support package must be linked to the code in order for it to run. Datasoft gives you choice of two different Run-time packages, a high speed integer version and a slower version that supports the full floating point functions.

The compilation procedure for the Datasoft compiler is fairly involved, after specifying the source and target filenames. The program asks the user to select the type of run-time package of either the high-speed integer or the slower floating point version. The appropriate Run-time package is then linked to your code, the compiler then studies your Atari Basic program and converts it into one or more Assembly language compatible files, which are written to disk.

Next the Datasoft system loads a three pass assembler, which reads the intermediate files created by the compiler and produces an assembly language binary object file which is the final executable version of your Atari Basic program. All assembly source files remain intact on disk, and may be accessed by Datasoft's 'DATASM Editor/Assembler (sold separately)' for later tweaking by hard-core hackers.

Datasoft's product is tricky to utilize if you only have access to just one disk drive, because the assembler and output files must be written onto the same disk as your Atari Basic source program. You have to be sure to leave enough space for them, according to Datasoft, this limits the maximum size of your Atari Basic program to about 100 sectors or 12.5k. Users with more than one disk drive can lessen the limitation by putting the Atari Basic source program on a separate disk.

GOOD DIAGNOSTICS:

An interesting feature of the Datasoft compiler is the Line Reference Map, this function displays each line number of your original Atari Basic program along with the exact address where it's machine language counterpart can be found. The map can be sent either to the screen immediately, a printer or a disk-file for later reference. Line references are very useful if you wish to de-bug or and modify the compiled version of your Atari Basic program.

The error handling of the Datasoft system is also helpful, problems that occur during the execution of a compiled program produce a standard Atari error number along with the address of the instruction that caused the foul up. If you prepared a line reference map of the program, you can determine which line in your Atari Basic program produced the error, the Datasoft system also allows you to restart a crashed program at any point by specifying a new run address.

THE CATCH:

It would be wonderful if you could take any old Atari Basic program, send it through one of these compilers and get a nice, speedy output file. Unfortunately, things are not that simple, both Monarch and Datasoft's products impose restrictions on the type of Basic code that can be successfully compiled.

Listings 1 and 2 show the documented programming restrictions of ABC and Datasoft Basic compiler respectively. Notice that the program access commands such as 'LOAD, SAVE, ENTER and LIST' are not supported by either system.

This makes sense because of the self-standing nature of a compiled program, also note that the floating point maths functions such as 'COS, SIN, etc,' cannot be used in either ABC or the integer version of the Datasoft compiler.

The documentation provided with ABC suggests a number of sneaky ways to get around it's lack of floating point arithmetic. It gives examples of how to simulate fractions, trigonometry and the RND(0) function without producing a compilation error. ABC's 24-bit integer maths package allows a usable variable range of either plus or minus of 8 million, so it's possible to scale almost any value to a convenient whole number.

Both integer and floating point versions of the Datasoft compiler offer a nice implementation of the RND(0) function, it also allows you to utilize the RUN command as long as you don't include a filespec such as 'RUN "D1:PROGRAM2.EXT"'.

Datasoft's compiler wont let you use variable as line references such as '10 X=30: Y=100:', '20 GOSUB X: GOTO Y+X*3' etc. Neither can you embed DATA statements in your Atari Basic code, you have to place them all at the very end of your program, preceded by either an 'END, STOP, GOTO #' statement. I personally like to keep DATA statements close to the corresponding READ statements, because it makes the programs more easier to de-bug. I also like to use variables as line references, because it makes my code self-documenting statements like 'GOSUB NEWLINE' more meaningful than 'GOSUB 2011'. Hopefully a later version of the Datasoft compiler will deal with this common stylistic approach more realistically.

WHICH ONE IS FASTER:

Speed is one of the main reasons for using an Atari Basic compiler. To compare the speed performance of the Monarch and Datasoft products, I wrote a short benchmark program that uses nested FOR/NEXT loops to fill a graphics 24 screen with direct POKE's (see listing 3:). The hardware timers at locations 19 and 20 keep track of the execution speeds in 60th's of a second or what is known as jiffies.

The benchmark program was compiled and executed on a standard 48k system, using 'Atari Basic, Atari Microsoft Basic v1.0 (disk), OSS Basic A+ v3.05 and of course ABC and both version of the Datasofts compiler'. The benchmark program was run three times on each system, and the results were average, to produce the results in listing 4:.

The 5-20 times speed improvement claimed by Datasoft's integer compiler was clearly justified. Similarly ABC speed increase is about 7.4 times faster, also right in line with Monarch's advertising. The floating point version of Datasoft's compiler is not very impressive in this example it was not much faster than OSS's Basic A+.

Prospective users should know that graphics statements such as 'PLOT, DRAWTO and FILL (XIO)' will not be significantly speeded up by using one of these compilers.

BASIC COMPILER continued

Since the rom routines that perform these functions are the same ones used by Atari Basic, it would be nice to see a super-compiler complete with it's own set of speedy graphics routines, similar to those offered by the 'Val-Forth system'.

MEMORY REQUIREMENTS:

The amount of memory required by a compiled Atari Basic program depends on three things: The size and type of program being compiled, the efficiency of the compilation, and the type and size of the Run-time package required to support the code.

ABC's Run-time package takes up 36 sectors or about 4.5k in ram, the floating point version of the Datasoft compiler requires 32 sectors or 4k in ram, but Datasoft's integer version just requires only 29 sectors or about 3.6k. These figures represent the minimum ram overhead required by any compiled program regardless of it's size or function.

We looked far and wide for a large Atari Basic program that could be used as the basis for a size comparison between the Datasoft's compiler and Monarch's ABC, Most of the trouble was caused by the Datasoft compiler, which would not accept embedded DATA statements found in virtually every off the shelf Atari Basic program we tried. In desperation, I re-wrote this issue's feature game (Adventure in the 5th dimension) without using variable GOTO's or GOSUB's, misplaced DATA lines or anything else that would violate the restrictions documented by either compiler.

After completely de-bugging the adventure, I saved it to disk and checked it's file size, the new version was just within the bounds of the maximum file size for a single drive on the Datasoft compiler with only 99 sectors. I then proceeded to compile the adventure with ABC with my single drive and 48k system. I experienced no problems until the very end of the compilation, it reported an error '166 point data length)?

This really puzzled me, so I called up Monarch and spoke to the author of the compiler, he tracked down the problem and found too many GOSUB's on line 66. He also suggested an easy fix and promised to eliminate the limitation in all future releases. My second compilation was flawless, the P-code produced by ABC was only 129 sectors in length, about 30% larger that the original Atari Basic program and the compiled adventure runs perfectly, but of course a lot faster.

Next I tried compiling the adventure with the Datasoft compiler, again using the same setup as ABC. I followed the instructions in the user's manual and copied the system equates file to the same disk as my Atari Basic program, then I ran the compiler. Before the end of the first pass, the compiler reported an error '162 disk full'. I looked at the disk from DOS and found that the assembler files had completely filled the disk, leaving no room for the assembly itself.

So I borrowed another drive and recompiled, using a second disk containing copies of the assembler, equates file and Run-time library files, again I was greeted with an error '162 disk full'.

Not to be deterred I copied the 'ASM.OBJ' file onto the same disk as the adventure and tried one more time, at last success. The compiler barely found enough room to write the assembler files, and I finally made it through the first pass.

My disk space difficulty was caused by the fact that the Datasoft compiler always writes assembler files to drive 1:. The reference manual estimates that these files require at least about five time the size of your Atari Basic program. That places the maximum possible Atari Basic program at 141 sectors or about 17.6k in ram, regardless of how many drives you have or can borrow, based upon a single density 90k drive such as the Atari 810.

Now the compiler started on passes 2 and 3, but the compiler stopped again to tell me that I had some unresolved line numbers, but it did not specify which lines were causing the problems. So I checked carefully through the Atari Basic program for GOSUB's or GOTO's that used a variable instead of a line number, but I did find nothing wrong with them.

The RESTORE statements in lines 73 and 79 do use variable references, but the Datasoft's documentation does not says anything about RESTORE's. So I wrote a little test program to see if the compiler would accept RESTORE's with variables, and sure enough the test failed.

I consider this undocumented restriction to be a very serious 'read BUG'. Data line addressing is one of the most powerful features of Atari Basic, since I had used it extensively in the adventure program, because it made object handling so much easier. Re-writting the adventure was completely out of the question, so I compiled the program one last time and ordered the assembler to ignore the unresolved line numbers, the remainder of the compilation proceeded without any more errors. The final program size was 214 sectors or about 20.9k, more than twice the size of the original Atari Basic program. Due to the presence of known errors, I did not even try to run the compiled version.

Other bugs in the Datasoft Basic compiler have been discovered by users of the first release. I have personally verified difficulties, the TRAP statement and the VAL function, along with some other confusing problems with strings and numeric arrays. Datasoft are aware of these bugs and will hopefully offer updated disks to the original purchasers of the first release.

THE ENVELOPE PLEASE:

The choice between Monarch's ABC and Datasoft's Basic Compiler is not an easy one, each product has an unique personality that make it suitable for specific applications and programming styles.

DATASOFT'S COMPILER:

On the plus side, if ultra-high-speed is very important to you, then the machine code produced by the Datasoft's integer compiler is very tough to beat. Datasoft's is also the better choice if you want to play around with the compiled versions of your software and if you have to use transcendental maths. The Datasoft's floating point package offers a slow

BASIC COMPILER continued

On the minus side, Datasoft's product is very greedy with disk space and ram, you need at least two drives to compile anything except small programs. You will also have to put up with the alarming range of Basic programming restrictions. Before you go and buy the Datasoft's compiler, may I suggest that you check to make sure that you're getting a bug-free version.

MONARCH'S ABC COMPILER:

Monarch's ABC is not as picky about your source code as the Datasoft compiler. It will compile just about anything that does not use fractions, and it's wide usable number range, gives it a decided advantage when it comes to simulating the floating point operations at high-speed. The P-code produced offers a degree of software protection you cannot get with straight 6502 assembly language, last but not least in 1983 Monarch's ABC costs $30 less than the Datasoft product.

AND THEN THERE'S BASM:

You may be wondering why I have not mentioned BASM, the third Basic compiler listed at the beginning of this article. The reason is very simple, BASM is not really a Basic compiler at all, it's a Basic assembler, an entirely new programming concept for the Atari computer. It looks like Basic, but acts like assembly language.

Take a look at listing 5:, this is the BASM equivalent of the speed benchmark used to test other Basics and compilers. Notice that some of the program lines look like Atari Basic and others look like assembly language, the REM statements are included in those places where BASM differs significantly form Atari Basic.

BASM programs are composed with a text editor supplied with the package. The source file is then saved to disk and assembled into machine language. A very small runtime package is then linked to the code and your application is ready to run.

The BASM system understands a very usable subset of Atari Basic, along with a number of statements and conditions not found in Atari Basic (See listing 6:). Primitive commands like PEEK and POKE must be replaced with their assembly language equivalent, LDA (LoaD Accumulator) and STA (STore Accumulator). READ/DATA structures are implemented by using the 6502's X and Y registers as indexes.

BASM allows you to mix both Atari Basic and assembly freely where possible, even on the same logical line. This arrangement combines the simplicity of Atari Basic with the Power of straight 6502 assembly language in an ingenious manner.

Because BASM programs have an assembly like syntax, the efficiency of compilation is much greater than that of either ABC or version of the Datasoft compiler. Only the pure basic statements are compiled and the assembly language sections of code are incorporated into the program as in machine language. This means that the speed of a BASM compiled program can approach the limits of the hardware.

I compiled the program in listing 5 and obtained an execution time of just 18 jiffies. That's just less than a third of a second, this is 231 times faster than Atari Basic equivalent. Computer Alliance claims a more conservative speed improvement of up to 130 times that of Atari Basic.

NOT FOR BEGINNERS:

BASM is not as straightforward to use as both ABC and the Datasoft compilers. You will have a hard time following the 72-page reference manual, unless you already know something about the 6502 architecture and assembly language programming. It took me a while to grasp the syntax required for certain types of Basic variables and addressing modes. More complete documentation is definitely called for, even if it means raising the price a bit.

I also ran across a bug in the disk interface, my review copy of BASM bombed out whenever I tried to execute a compiled program more than once, this made it impossible to repeat my benchmark test without completely re-booting the system. When Computer Alliance fixes this problem, they will have a very impressive and very powerful Basic compiler.

IMPLICATIONS:

A stigma against Basic programming has arisen in the software market over the last few years. The prejudice is based on the absurd idea that the quality of a program has something to do with the language it was created on.

The compilers reviewed in this article will help Basic programming become more respectable again. For this reason alone I think that they are the most important pieces of Atari software to come down the pipe since Val-Forth.

They may actually be more significant, because they offer much of the performance of Forth without the need to learn a new programming language. That means Atari Basic programmers can spend less time puzzling over stacks, disk screens and all other unfamiliar concepts, and more time improving the performance and the quality of their Atari Basic software.

I'm quite happy to report that zero of these compilers mentioned in this article requires a licensing fee. You can sell your compiled software royalty-free as long as you include a credit in your documentation or program to say it was compiled on such a compiler and the contact address.

Atari basic compilers are about to open up the world of professional software developments to a whole new range of talented authors.

Let's hope the code they produce is as valuable as these three products.

LISTING 1: ABC's Programming Restrictions.

Unsupported functions:
ATN, CLOG, COS, EXP, LOG, RND, SIN, SQR.

BASIC COMPILER continued

Unsupported Arithmetic operators: ^ (exponentiation)

Unsupported statements:
BYE, CLOAD, CONT, CSAVE, DEG, DOS, ENTER, LIST, LOAD, LPRINT, NEW, RAD, RUN, SAVE.

Other restrictions:
Cannot use fractional (non-integer) values.
Cannot use constants greater than 65535.

LISTING 2: Datasoft's Programming Restrictions.
Unsupported functions: (integer mode only)
ATN, CLOG, COS, EXP, LOG, SIN, SQR.

Unsupported Arithmetic operators:
none.

Unsupported statements:
BYE, CLOAD, CONT, CSAVE, DOS, ENTER, LIST, LOAD, NEW, RUN "dev:FILENAME.EXT", SAVE.

Other restrictions:
Integer mode values limited to either plus or minus of 32767, (except address constants).
DATA statements must be at the end of the program and cannot be executed (see text).
DIM statements cannot use variables for size allocation (such as DIM X$(A)).
GOTO's and GOSUB's cannot use variables for line references (such as GOSUB FILE: GOTO MENU).

LISTING 3: Benchmark Program 1 for Atari Basic and Compilers.

```
10 REM ****************************
15 REM * BENCHMARK TEST FOR BASIC *
20 REM *        COMPILERS        *
25 REM ****************************
30 POKE 19,0:POKE 20,0
35 GRAPHICS 24
40 SETCOLOR 1,0,14:SETCOLOR 2,0,0
45 SCREEN=PEEK(88)+256*PEEK(89)
50 FOR I=0 TO 191: FOR J=0 TO 39
55 POKE SCREEN+J,255
60 NEXT J: SCREEN=SCREEN+40: NEXT I
65 GRAPHICS 0
70 PRINT PEEK(20);" Jiffies"
75 PRINT PEEK(19);" Jiffies x 256"
```

LISTING 4: Speed Test Results from Program 1:.

Run-time m	Jiffies	Hours:Minutes:Seconds.100's
Atari Basic rev (A) rom	4160	00:01:09.30
Atari Microsoft Basic v1.0	3348	00:00:55.80
OSS Basic A+ v3.05	2717	00:00:45.30
Monarch's ABC compiler	565	00:00:09.40
Datasoft's FP compiler	2435	00:00:40.60
Datasoft's integer compiler	218	00:00:03.60
(Note BASM results were produced with listing 5: program 2:		
Computer Alliance's BASM	18	00:00:00.33

LISTING 5: Benchmark program 2: for Computer Alliance's BASM.

```
0100 REM * PROGRAM EQUATES
0110SCREEN=88
0120TIMER=20
0130TIMER256=19
0140 REM * POKE 19,0: POKE 20,0
0150 LET TIMER256=0:LET TIMER=0
0160 GRAPHICS 24
0170 SETCOLOR 5,0,14:SETCOLOR 6,0,0
0180 FOR I=0 TO 191:FOR J=0 TO 39
0190 REM * POKE SCREEN+J,255
0200 LDA #255:LDY J:STA (SCREEN),Y:NEXT J
0210 REM * SCREEN=SCREEN+40
0220 REM * THIS IS A 16-BIT BINARY ADDITION
0230 CLC:LDA SCREEN:ADC #40:STA SCREEN+1
0240 LDA SCREEN+1:ADC #0: STA SCREEN+1
0250 NEXT I
0300 REM * GRAPHICS 0
0310 FILE 0
0320 BPRINT TIMER:PRINT " Jiffies"
0330 BPRINT TIMER256:PRINT " Jiffies x 256"
0340 RETURN : REM * BACK BASM
0350 REM * Line 360 initializes the variable I,J
0360 DIM I,J
```

LISTING 6: Computer Alliance's BASM keywords.

BINPUT, BPRINT, CLOSE, COLOR, DATA, DEF, DIM, DRAWTO, ENDDEF, FILE, FILL, FOR, GET, GOSUB, GOTO, GRAPHICS, IF, INPUT, LET, LOCATE, NEXT, OPEN, PLOT, POSITION, PRINT, PUT, REM, RETURN, SETCOLOR, SOUND, STOP, TR, TRAP, WHILE.

Supplied, typed in, tidied up and edited by:

Mr. Andrew MCINTOSH
21 ROMAN ROAD
BONNYBRIDGE
STRIRLINGSHIRE
FK4 2DE, SCOTLAND

Level Codes for Ke-Soft's

LASERMAZE:

1 LASER	2 HYPER	3 SPACE	4 DIGIT
5 TUNED	6 ATARI	7 MARIO	8 TECNO
9 SOGON	10 BASIC	11 LEVEL	12 HODUL
13 HOOCH	14 HONEY	15 ELEGY	16 DEATH
17 CACAO	18 CABAL	19 BIGOT	20 AGAIN
21 HAITI	22 INDIA	23 JESUS	24 KOREA
25 CHINA	26 WUSHU	27 MONTY	28 NANCY
29 CAMEL	30 SARAH	31 WALES	32 TIMES
33 WHIZZ	34 TITAN	35 SYNTH	36 STORM
37 SHAVE	38 SHARK	39 ROUTE	40 PIECE
41 PINCH	42 OSCAR	43 OTTER	44 MAJOR
45 LOWER	46 LUCID	47 KNAVE	48 LABEL
49 INPUT	50 SUPER		

SEARCHING FOR BOBBY FISCHER ON THE 8-BIT ATARI:

An Annotated List of Available Programs

By Bill Hall, Canada.

Computer Chess

The earliest of all chess programs for the 8-bit Atari, this 1979 cartridge-based game from Atari Corp. sports bilious colours, abysmal graphics, and almost no features. Joystick-operated.

Chess 7.0

By Larry Atkin (Odesta 1982). The best of the early games with many features and serviceable graphics. Cursor movement is rather awkward, accomplished by paddles or cursor keys. Translator required.

Micro Chess

A simple version with odd graphics and few features. Movement by algebraic notation. Translator required--specifically, the Atari OS translator by Jonathan Sanders.

Sargon

Written by Dan and Kathe Spracklin, this program is one of the oldest, has gone through several incarnations, and was available on many different computers. The original Sargon was written in Z-80 code and analyzed in the book, "Sargon--A Computer Chess Program." The publisher, Hayden, also released the subsequent software versions. Sargon 2 (1982) employs algebraic notation (though coordinates are not displayed) and remarkably ugly graphics. Sargon 3 (1985) is a much improved version, and for a time was considered the front-runner in the chess wars. The Atari version was done by Lynn and Alex Ford. Decent graphics, and movement is accomplished by joystick or algebraic notation.

Chess

This cartridge-based game from Parker Brothers offers bright, pleasant graphics, and just enough features to satisfy those players without a deep interest in chess. It is often found in PD libraries and, though it lacks any distinguishing text, it can be identified by its length (66 sectors). An interesting aside: this is the only Atari 8-bit game which shows pieces actually moving from one position to another. In all other games a piece is "moved" by disappearing from its old position and reappearing at its new position. Joystick only.

Chess

Rather surprisingly, this program by John Krause is the only BASIC chess program for Atari 8-bit.

ULTRASONIC SWITCH

By Len Golding

Ever since man first poked his enemy with a stick, the subject of remote control has fascinated the human mind. You can switch on your TV or VCR, open your garage doors, lock your car, drive model aircraft and a host of other clever things without stirring from the spot. But this armchair revolution seems to have by-passed the computer world - certainly the Atari 8-bit machines have a lot of catching up to do. So here's a gadget to start us on the right road - it's a simple remote control device which lets you send messages to your computer from a distance of up to six metres.

It can add an extra dimension to audio-visual displays in schools or staff-training centres, allowing you to step through a pre-set sequence of stills or computer-animated drawings at the touch of a button, without any trailing wires. It can also be used to trigger any of the power-control gadgets described in previous issues, so you could switch mains or battery-powered appliances on or off without going near them. It can even be made to operate as a stand-alone unit, to switch appliances on or off without the help of your computer.

We've chosen an ultrasonic system, for various reasons. It's cheaper and simpler than radio control, and beats infra-red on range, cost of operation and susceptibility to extraneous factors in the outside world. There's also no risk that it will accidentally change channels on your IR-controlled TV or activate your VCR. It uses a single on/off channel, but this is adequate for many purposes, as we'll explain later.

Fig 1 shows the transmitter. It's a simple oscillator which can be adjusted by VR1 to generate any frequency from around 35Khz to 45KHz, so it can be tuned precisely to the ultrasonic emitter's optimum frequency. Fig 2 shows the receiver, which is a little more complex.

TR1 and TR2 form a simple but powerful amplifier, which converts the tiny output from the ultrasonic transducer into a respectable voltage swing at the input of IC2. This IC is a phase-locked loop device, whose output goes low whenever a signal of the correct frequency is received. It serves three purposes: first as an audio-to-digital converter, second as a current amplifier and third as a filter which rejects any spurious incoming frequencies.

Construction of both boards is easy, especially if you use the ready-made PCBs available from RH Design. Alternatively you may prefer to etch your own from the patterns at Figs 3a and 3b.

The transmitter board layout is given in Fig 4. It's designed to fit into a small case supplied by Maplin Electronics, but could easily be adapted to suit other boxes if necessary. The switch is a high-quality push-button type, which is sturdy enough to hold the entire board firmly in place, but we've included four mounting holes in case you want to use a different switch or different box.

Make sure that IC1 is inserted with pin 1 in the correct position, but all the other transmitter components can go either way round.

ULTRASONIC SWITCH continued

TD1 is the ultrasonic transmitter, and is marked with a "T". You can attach it directly to a 2-way PCB terminal block as shown in fig 4, by bending its leads slightly. Alternatively, solder on longer leads as shown in the photograph - this will allow for adjustment of its position in the hand-held box. The battery leads must be soldered to veropins, or directly to their pads.

The receiver board layout is shown in Fig 5. The only polarised components are TR1, TR2, IC2 and C2. The transistors will fit only one way round; IC2 has a small notch in one end, as shown in Fig 5, and C2 has black chevrons to mark the 0v side. All other components can go either way round.

As with the transmitter board, the ultrasonic transducer TD2 can be wired to a two-way terminal block, or directly to the PCB pads via extension leads.

When everything has been correctly soldered, the two units must be tuned to work together. This can be a bit fiddly if you don't possess an oscilloscope, but it's worth the effort to achieve maximum reliability and range.

Start by plugging the receiver into port 1, and run Program 1. You should see the number "1" at centre screen. If you get '0', adjust VR2 until the number changes and holds there reliably. If you can't get a '1' to appear, there's a mistake in assembly somewhere.

Attach a 9v battery to the transmitter board and adjust VR1 to somewhere near its centre point. Hold it pointing at the receiver about half a metre away, press the button and adjust VR2 on the receiver board until the number on screen changes from 1 to 0, and the text window disappears. This adjustment is fairly critical. Now stand about two metres away and press the transmit button again. If the number on screen does not change, adjust VR1 on the transmitter until it does. Repeat this dual-adjustment procedure at about six metres distance, to get the best possible response.

Programming could hardly be simpler. Your computer thinks the switch is a joystick trigger button, so the number in address 644 changes from 1 to 0 whenever the unit is activated. You can use this to initiate a single action, or build a step counter into your program which selects options in sequence when the button is pressed. In this case it's a good idea to build in some kind of delay as well, to avoid triggering unwanted options on the way through the sequence.

Program 2 shows a suitable system. Hold down the Transmit button to step through the various options without triggering any of them, and release it when you get to the one you want. As it stands, it simply notifies you that a particular option has been triggered, but you can insert extra code at line 110 to make it do something useful. For example, it could trigger one of the power-control devices we've described in earlier issues, to control models or mains appliances. The power controller could plug into port 1, leaving port 2 for the ultrasonic switch. In this case the address controlled by the switch will be 645. Software to drive the power controllers themselves is described in the constructional articles for each device.

If you would prefer to use the remote switch as a stand-alone module, independent of your computer, attach a 6v or 9v battery in place of joystick leads 7 and 8 (+ve to the pin 7 point), and replace R9 with a 1N4148 signal diode, wired with its cathode (coloured band) towards IC2. Choose a 6v relay whose coil is 80 ohms or more, such as Maplin types FX88V, FM91Y, FM89W or FX23A, and connect it between the 'pin 7' and 'pin 6' terminals. The relay contacts will close when the transmit button is pressed and open again when it is released.

PARTS REQUIRED FOR ULTRASONIC SWITCH

TRANSMITTER

Maplin Order

		Codes
R1	1K (brown/blak/red)	M1K
R2	12K (brown/red/orange)	M12K
VR1	4K7 Horizontal preset	UH02C
C1	.001 mfd Mylar (marked 102K)	WW15R
S1	Push-to-make switch	YR67X
IC1	TLC555C	RA76H
TD1	Ultrasonic transmitter *	
	PP3 battery lead	HF28F
	PP3S battery	FK62S
	8-pin DIL socket	BL17T
	Small narrow box (optional)	FT31J

RECEIVER

R3, R6	1M (brown/black/green)	B1M
R4	47K (yellow/violet/orange)	M47K
R5	1K (brown/black/red)	M1K
R7	10K (brown/black/orange)	M10K
R8	18K (brown/grey/orange)	M18K
R9	4K7 (yellow/violet/red)	M4K7
VR2	10K horizontal preset	UH03D
C2	100mfd 10v single-ended	FF10L
C3	3n3 ceramic	WX74R
C4	22n ceramic	WX78K
C5	1n0 ceramic	WX68Y
C6	0.1mfd disc ceramic	YR75S
C7	0.22mfd Mylar (marked 224K)	WW83E
C8	0.01mfd Mylar (marked 103K)	WW18U
TR1,TR2	BC109C	QB33L
IC2	NE567 tone decoder	QH69A
TD2	Ultrasonic receiver *	
	3-way PCB terminal	RK72P
	2-way PCB terminal (optional)	FT38R

* Note: the ultrasonic transducers are sold as a pair, order code HY12N

All components are available from:
Maplin Electronic Supplies, P.O. Box 3
Rayleigh, Essex, SS6 8LR
Tel: 0702 552911

Printed circuit boards (order code DBP12a and DBP12b) Price £2.38 per pair.

Joystick extension lead (order code AT111) Price £2.99
Available from:
R.H. Design, 137 Stonefall Avenue,
Harrogate, North Yorkshire,
HG2 7NS Tel: 0423 508359

ULTRASONIC SWITCH continued

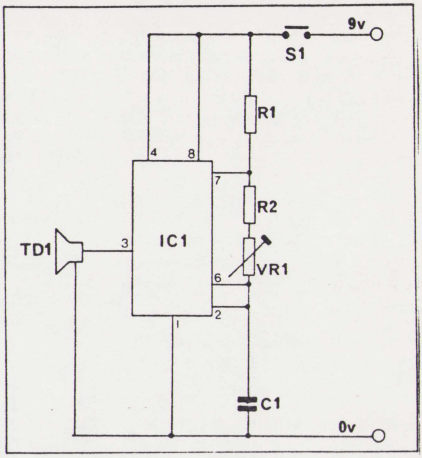


Figure I: Circuit for transmitter

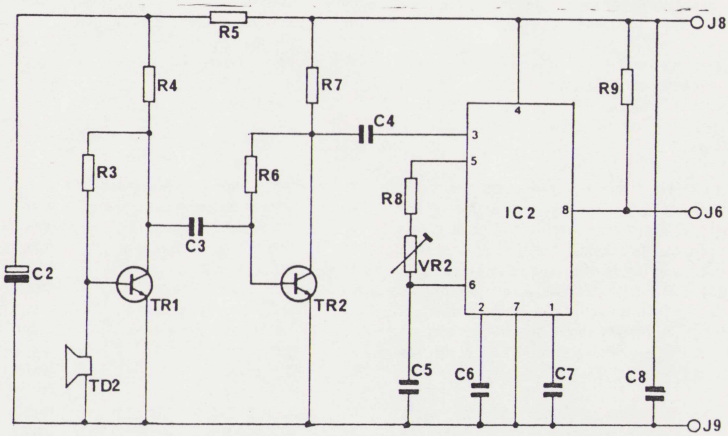


Figure II: Circuit for receiver

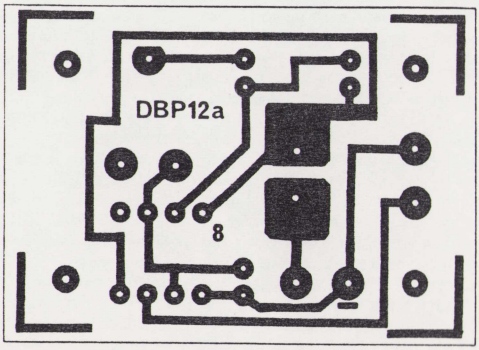


Figure IIIa: The PCB transmitter pattern

ULTRASONIC SWITCH continued

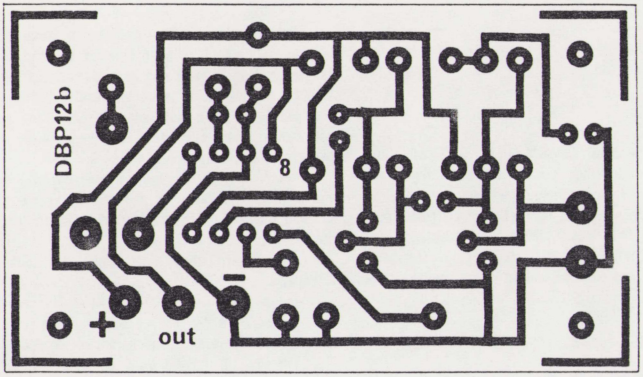


Figure IIIb: The PCB receiver pattern

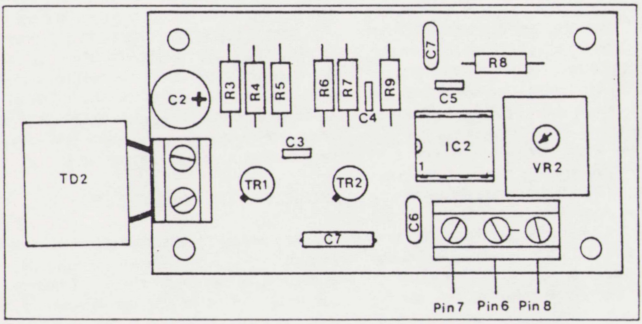


Figure V: Component layout for receiver

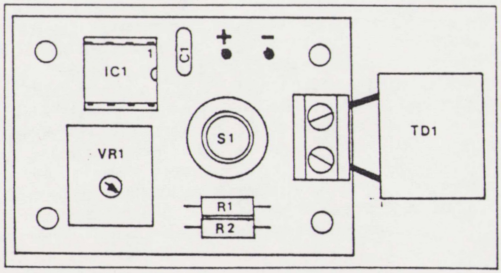


Figure IV: Component layout for transmitter

CRACKING THE CODE

by Keith Mayhew

Re-printed by M. Gerum

This article first appeared in "The UK ATARI Computer Owners Club" later renamed "MONITOR"

Part 13

The principles of smooth scrolling,both vertically and horizontally, were described last time. The most important fact is that scrolling can be achieved without the need to move any display data, thus making it fast. We continue with some example programs that demonstrate how display lists can be built for scrolling characters; such methods form the basis of many games but are also of use in other applications such as word processors.

VERTICAL SCROLLING

Any display list instruction which has its vertical scroll flag bit set will cause its associated line on the display to be shifted upwards by the number of scan lines specified by the contents of the VSCROL register.

When a group of consecutive lines have their vertical scroll flag bits set then the whole region will move together as VSCROL is altered. Such a region is terminated by the first line which does not have its vertical scroll flag bit set. The last line actually forms part of the scrolling region. So the smallest possible region must consist of two lines; one with its vertical scroll flag bit set and the next which does not have it set. The last line in any region is displayed as only one scan line in height.

When VSCROL has the value zero then a region is displayed in its normal position with only the top scan line of the last line visible. As VSCROL is increased in value, the region is moved up, losing scan lines from the top of the first line, while bringing in data from the last line. Note that no matter what value VSCROL contains, the region remains exactly the same number of scan lines in height.

Figure 1 illustrates a scrolling region of text and the corresponding display image produced for different values of VSCROL, assuming that each line is normally eight scan lines in height.

Once a region has been scrolled upward by its maximum amount using VSCROL, i.e. it is one scan line away from the next line's normal position, then VSCROL should be set back to zero whilst a coarse scroll is performed by adjusting the LMS addresses by adding the number of bytes for one line of data, eg. 40 bytes for BASIC'S GRAPHICS 0 mode. Scrolling the image in the opposite direction is simply a case of decrementing VSCROL and then resetting VSCROL to its maximum value and subtracting from the LMS address. The LMS instruction in the display list will usually be placed on the first scrolling line of a region, determining where the display data will come from.

Listing 1 gives an example of a vertically scrolling region in a display. Listing 2 is the BASIC program to read in the code; after running it, the code can be executed by typing:

The display will show the contents of memory from location 0. By using a joystick the display can be moved up or down over memory while the top line of the display will indicate the address of the first memory location being viewed.

The display list is defined on lines 1900 to 1970 of Listing 1. It consists of a line at the top which displays the text "Location; 0000" and ends with a line which displays the text "Use joystick to move vertically.". The scrolling lines in the middle of the display list have not been defined but space has been reserved for them to be filled in by the program when it is executed.

After turning off the screen DMA, the program calls on the routine "CHCONV" to perform the conversion between the ATASCII codes generated by the assembler to display the codes of the character set (as explained last time). The new display list is then installed and some variables are initalised.

The missing part of the display list is now built with an LMS and address for each line. The addresses are initialised so that each points 32 bytes further on from its predecessor.

The DMA is turned back on with the screen set to the narrow size, which gives 32 bytes per mode 2 line, the VBI routine is enabled and control is returned back to the caller.

During each VBI the joystick is tested to see if it has been moved up or down. If movement has occurred then the scrolling technique described above is used. Note that a copy, or shadow, of the VSCROL register is kept in "COUNT"; this is because the scroll registers are write only.

When a coarse scroll is to be performed then all of the LMS addresses are updated by either adding or subtracting 32. After a coarse scroll the routine "PRADDR" is called to print the new address of the first LMS at the top of the scren as a four digit hex string. "PRADDR" uses "PRHEX" to print each of the two bytes which then calls on "PRDIGIT" to print each hexadecimal digit to the screen.

DISPLAY BOUNDARIES

Just one LMS instruction could have been used for the scrolling region but this would clearly have caused a problem as the display can cross over a 4 K boundary. If the display was to be limited to only one 4K block at a time then this approach would be suitable. However, because we want a continuous display, regardless of boundaries, it is necessary to use an LMS instruction on every line.

Choosing the narrow width screen DMA in this example means that each mode 2 line requires 32 bytes. The consequence of this is that multiples of 32 give multiples of 4096, i.e. 32 divides exactly into 4096; so any boundary will always occur at the start of a line. As every line reloads the LMS with the address of the next line, the boundary has effectively been lost!

If a standard width screen was used then, at 40 bytes per line, boundaries would rarely occur at the start of a line and, so we wish to display memory contiguously from line to line, there is no way of avoiding boundary problems. Similary a wide screen at 48 bytes per line, would also cause problems with boundaries.

Does this mean that to scroll correctly over a 4K byte boundary that a narrow screen must be used? The answer is no; the reason why the wider screens could not be used in our example is that contiguous memory is being displayed

CRACKING THE CODE continued

from line to line, however, if we use noncontiguous memory for the display then ths problem is easily overcome.

There are two basic approaches which can be used to display a continuous image whilst crossing boundaries.

The most obvious method is illustrated in Figure 2, this has continuous image data starting at the first boundary and extending to just before the next boundary. Once the screen has been scrolled into the last part of this image then the display is switched to the start of the next boundary. By having a copy of the previous screen's image at the start of the next block means that the display will appear continuous as the next block is brought into view. The advantage of this method is that only one LMS instruction is needed but approximately one screen's image data has to be duplicated for each new block.

The second method requires LMS instructions on every line but each picks up data spaced at say 64 bytes apart so that any boundary will always coincide with the start of a line. Figure 3 illustrates how the display image comes from the first part of each line with some unused memory at the end of each line. Unlike the previous method, the data has to be noncontiguous because of the unused memory on each line.

HORIZONTAL SCROLLING

As with vertical scrolling, having the horizontal flag bit set for a line, causes it to be moved to the right by the number of color clocks specified by the contents of the HSCROL register. Note that as scrolling is done by color clocks, the HSCROL register only extends from 0 to 3 for a standard character cell, such as mode 2, not 7!

Once a line has been shifted by its maximum amount then HSCROL should be reset to zero and one added to the address of the LMS instruction for that line. Reversing this process produces scrolling to the left instead of the right. To scroll a region of lines horizontally requires each to have its horizontal scroll flags set and each to have its own LMS instruction.

Figure 3, which shows the set-up for a vertically scrolling screen, also serves as an illustration of horizontal scrolling over several lines; the line length might be anything up to 4K bytes. As with vertical scrolling it is necessary to arrange data so that boundaries occur at the start of a line.

Listing 3 is a program which sets up a display list to scroll a single line of mode 2 characters in a continuous loop like a message board. After adjusting the text string for display, the first 44 characters are copied to just after the end of the message string; this corresponds to the data duplication for vertical scrolling as in Figure 2.

Once the VBI has been installed the message is fine scrolled to the left. The coarse scrolls are counted by the variable "POS". When "POS" reaches its maximum value it means that the whole message has been displayed and the copy of the start of the message is now in view. The LMS address is reset at this stage to point back to the start of the string which is identical to the image already being displayed - so the loop can continue, making the message look infinitely long. Listing 4 is the BASIC program to read in the code for this program, it is executed by typing:

```
X=USR(24576)
```

SCREEN WIDTHS

The bottom two bits of DMACTL determine the width of the screen, i.e. how many bytes will be fetched for a line of data. The narrow screen (bit 1 set to 0 & bit 0 set to 1) has a total of 128 color clocks in width. The standard screen (bit 1 set to 1 & bit 0 set to 0) has 32 more color clocks, giving it a total of 160. Lastly, the wide screen (bit 1 set to 1 & bit 0 set to 1) has a further 32 color clocks, giving it a total of 192.

This information is useful in the context of horizontal scrolling because ANTIC actually fetches different amounts of data for a line when its horizontal scroll flag bit is set. For a narrow screen the amount of data fetched for a scrolling line is the same as that for a standard screen. A standard screen uses the same amount as a wide screen. A wide screen, however, does not cause any extra data to be fetched for a scrolling line.

For the narrow and standard screen widths, scrolling lines take their data from 16 color clocks further in than usual. Thus you must arrange your data accordingly. As HSCROL is increased in value then this hidden data at the left edge of the screen starts to become visible.

In wide screen mode the situation is much simpler. There are no offsets for data to be accounted for as data is displayed as normal. As HSCROL is increased in value then background color is scrolled in to fill up the gap at the left edge. This is really of little consequence, though, as the extremes of the picture to the left and right are usually well out of visibility on most monitors and televisions.

WHAT NEXT

Having seen how horizontal and vertical scrolling can be used all that remains is for you to make use of these techniques in your programming.

It should be obvious that horizontal and vertical scrolling can be mixed at will and it is very easy to produce a screen which scrolls in all directions over a background, or landscape, which is used as the basis to many games. Of course, multicolour character sets make a world of difference over boring old mode 2!!! The use of DLI's to produce areas scrolling at different speeds or directions is also quite effective. Just add some players and missiles and you will have exploited much of the power of these fine machines. Above all experiment; it is the only real way to learn!

NEXT TIME

Yes, this series will undoubtedly be back next time!.

Over thirteen parts we have covered just about everything from binary arithmetic to a detailed look at the hardware features for multicolour graphics.

In the future we will still have some more mundane aspects of the hardware to cover as well as the use of the sound channels. However, the main area of discussion is going to be the operating system.

Editors note: In part 12 of this series the word "Luminescence" was inadvertently substituted for "luminance", a small but significant error, sorry for any misunderstanding it may have caused.

CRACKING THE CODE continued

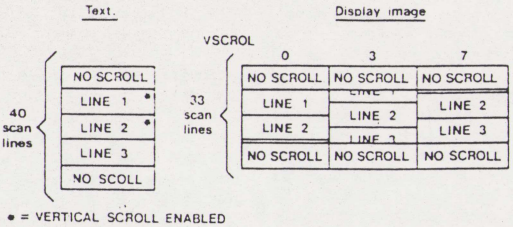


Figure 1.

```
0100 ;Demo of smooth vertical scrolling.
0110 ;Hardware registers...
0120 DMACTL  =      $D400   ;DMA control.
0130 VSCROL  =      $D405   ;Vertical scroll.
0140 ;Operating system shadows...
0150 SDMCTL  =      $022F   ;DMA control.
0160 SDLSTL  =      $0230   ;Display list pointer low.
0170 SDLSTH  =      $0231   ;Display list pointer high.
0180 STICK0  =      $0278   ;Joystick 0.
0190 ;Operating system vectors...
0200 SETVBV  =      $E45C   ;Set VBI vector.
0210 VBI2    =      $E45F   ;VBI stage two vector.
0220 ;Display list equates...
0230 LINES   =      48      ;Number of display lines + 3.
0240 JMP     =      $01     ;Jump.
0250 WVB     =      $40     ;Wait for vertical blank flag.
0260 LMS     =      $40     ;Load memory scan flag.
0270 VS      =      $20     ;Vertical scroll flag.
0280 B4      =      $30     ;Blank 4 lines.
0290 3B      =      $70     ;Blank 8 lines.
0300 ;Page zero variables...
0310         *=     $CB
0320 PZERO   *=     *+2     ;Pointer.
0330 COUNT   *=     *+1     ;Scroll counter.
0340         *=     $4000
0350         PLA
0360         LDA    #$00    ;Turn off DMA.
0370         STA    SDMCTL
0380         STA    DMACTL
0390         LDY    #48     ;Correct 48 bytes of text.
0400         LDA    #TITLE&$FF
0410         STA    PZERO
0420         LDA    #TITLE/256
0430         STA    PZERO+1
0440         JSR    CHCONV
0450         LDA    #DLIST&$FF ;New display list.
0460         STA    SDLSTL
0470         LDA    #DLIST/256
0480         STA    SDLSTH
0490         LDX    #0      ;Zero scroll counter.
0500         STX    COUNT
0510         STX    VSCROL
0520         STX    PZERO   ;And LMS pointer.
0530         STX    PZERO+1
0540 LOOP    LDA    #LMS+VS+2 ;Save an instruction.
0550         STA    DATA,X
0560         LDA    PZERO   ;And its address.
0570         STA    DATA+1,X
0580         LDA    PZERO+1
0590         STA    DATA+2,X
0600         LDA    PZERO   ;Move pointer to next line.
0610         CLC
0620         ADC    #32
0630         STA    PZERO
0640         BCC    SKIP
0650         INC    PZERO+1
0660 SKIP    INX            ;Next LMS instruction.
0670         INX
0680         INX
0690         CPX    #LINES-3
0700         BNE    LOOP
0710         LDA    #LMS+2  ;Last one has no scroll flag.
0720         STA    DATA,X
0730         LDA    PZERO
0740         STA    DATA+1,X
0750         LDA    PZERO+1
0760         STA    DATA+2,X
0770         LDA    #$21    ;DMA on with narrow screen.
0780         STA    SDMCTL
0790         LDY    #VBI&$FF ;Install VBI.
0800         LDX    #VBI/256
0810         LDA    #6
0820         JSR    SETVBV
0830         RTS
0840 ;Vertical blank interrupt.
0850 VBI     LDA    STICK0  ;Test for up or down movement.
0860         AND    #1
0870         BEQ    UP
0880         LDA    STICK0
0890         AND    #2
0900         BEQ    DOWN
0910         JMP    VBEXIT
0920 UP      DEC    COUNT   ;Up - decrement scroll count.
0930         BPL    UFINE   ;If positive then fine scroll.
0940         LDA    #7      ;Reset count.
0950         STA    COUNT
0960         LDX    #0      ;Adjust all LMS bytes.
0970 UPLOOP  LDA    DATA+1,X
```

CRACKING THE CODE continued

```
0980          SEC
0990          SBC     #32
1000          STA     DATA+1,X
1010          BCS     USKIP
1020          DEC     DATA+2,X
1030 USKIP    INX
1040          INX
1050          INX
1060          CPX     #LINES
1070          BNE     UPLOOP
1080          JSR     PRADDR  ;Print new LMS address.
1090 UFINE    LDA     COUNT   ;Update fine scroll value.
1100          STA     VSCROL
1110          JMP     VBEXIT  ;Exit.
1120 DOWN     INC     COUNT   ;Down - Increment fine count.
1130          LDA     COUNT
1140          CMP     #8
1150          BNE     DFINE   ;If not max. then fine scroll.
1160          LDX     #0      ;Reset fine scroll count.
1170          STX     COUNT
1180 DNLOOP   LDA     DATA+1,X ;Adjust all LMS bytes.
1190          CLC
1200          ADC     #32
1210          STA     DATA+1,X
1220          BCC     DSKIP
1230          INC     DATA+2,X
1240 DSKIP    INX
1250          INX
1260          INX
1270          CPX     #LINES
1280          BNE     DNLOOP
1290          JSR     PRADDR  ;Print new LMS address.
1300 DFINE    LDA     COUNT   ;Update fine scroll.
1310          STA     VSCROL
1320 VBEXIT   JMP     VBI2    ;Back to O.S.
1330 ;Print out address of first LMS.
1340 PRADDR   LDA     #NUMBER&$FF ;Copy address of number.
1350          STA     PZERO
1360          LDA     #NUMBER/256
1370          STA     PZERO+1
1380          LDY     #0      ;Zero index for high byte.
1390          LDA     DATA+2  ;Get high byte.
1400          JSR     PRHEX   ;Print it.
1410          LDY     #2      ;Index to low pair digits.
1420          LDA     DATA+1  ;Get low byte.
1430          JSR     PRHEX   ;Print it.
1440          RTS             ;Return.
1450 ;Print a byte in hex format.
1460 PRHEX    PHA             ;Save A.
1470          LSR     A       ;Get high nibble.
1480          LSR     A
1490          LSR     A
1500          LSR     A
1510          JSR     PRDIGIT ;Print it.
1520          PLA             ;Restore A.
1530          AND     #$0F    ;Mask for low nibble.
1540          INY             ;Next character index.
1550          JSR     PRDIGIT ;Print it.
1560          RTS             ;Return.
1570 ;Print a single hex digit.
1580 PRDIGIT  CMP     #$0A
1590          BCC     DIGIT   ;Below 10 is digit.
1600          CLC             ;Else add to get char. code.
1610          ADC     #$17
1620          JMP     SAVCHR
1630 DIGIT    CLC             ;Digit - Adjust it.
1640          ADC     #$10
1650 SAVCHR   STA     (PZERO),Y ;Save character code.
1660          RTS             ;Return.
1670 ;Convert an ATASCII string to display characters.
1680 CHCONV   DEY             ;Y holds number of characters.
1690          LDA     (PZERO),Y ;Get character.
1700          TAX             ;Save it.
1710          AND     #$7F    ;Turn off high bit (inverse).
1720          CMP     #$60
1730          BCC     CH1
1740          TXA             ;Above $60 - no adjustment.
1750          JMP     CHOK
1760 CH1      CMP     #$20
1770          BCC     CH2
1780          TXA             ;Between $20 and $5F.
1790          SEC             ;Subtract $20.
1800          SBC     #$20
1810          JMP     CHOK
1820 CH2      TXA             ;Between $00 and $1F.
1830          CLC             ;Add $40.
1840          ADC     #$40
1850 CHOK     STA     (PZERO),Y ;Save character back.
1860          CPY     #$00    ;Last character?
1870          BNE     CHCONV
1880          RTS             ;Return.
1890 ;Display list data.
1900 DLIST    .BYTE   88,88,88,LMS+7
1910          .WORD   TITLE   ;Title string address.
1920          .BYTE   34
1930 DATA     *=      *+LINES ;Filled in by program...
1940          .BYTE   34,LMS+2
1950          .WORD   FOOTER  ;Footer string address.
1960          .BYTE   JMP+WVB
1970          .WORD   DLIST
1980 TITLE    .BYTE   " LOCATION: 0000 "
1990 NUMBER   =       TITLE+11 ;Address of first '0' in title.
2000 FOOTER   .BYTE   "Use joystick to move vertically."
```

Listing 1.

CRACKING THE CODE continued

```
QZ 10 DIM HEX$(16)
HX 20 LINE=10000:TRAP 100:J=0:START=24576
VA 30 READ HEX$,CHKSUM:SUM=0
AA 40 FOR I=1 TO 15 STEP 2
IG 50 D1=ASC(HEX$(I,I))-48:D2=ASC(HEX$(I+
   1,I+1))-48
KT 60 NUM=(((D1-7*(D1>16))*16+(D2-7*(D2>16
   )))
LW 70 SUM=SUM+NUM:POKE START+J,NUM:J=J+1:
   NEXT I
LY 80 IF SUM=CHKSUM THEN LINE=LINE+10:GOT
   O 30
IN 90 ? "Checksum error on this line:"
VO 95 LIST LINE:END
YS 100 PRINT "Data in memory."
RA 10000 DATA 68A9068D2F02BD00,604
KQ 10010 DATA D4A030A96C85CBA9,1202
VH 10020 DATA 6185CC200A61A92E,788
JT 10030 DATA 8D3002A9618D3102,649
ZY 10040 DATA A20086CDBE05D486,994
WF 10050 DATA CB86CCA9629D3561,1115
YI 10060 DATA A5CB9D3661A5CC9D,1202
QO 10070 DATA 3761A5CB1B692085,814
AC 10080 DATA C39002E6CCEBEBEB,1479
MA 10090 DATA E02DD0DFA9429D35,1145
NG 10100 DATA 61A5CB9D3661A5CC,1142
UG 10110 DATA 9D3761A9218D2F02,701
UW 10120 DATA A06AA260A906285C,823
LU 10130 DATA E460AD7802290 1F0,901
YX 10140 DATA 0AAD7802290 2F02D,633
BB 10150 DATA 4CCE60C6CD101EA9,996
KS 10160 DATA 0785CDA200BD3661,847
NB 10170 DATA 3BE9209D3661B003,808
EU 10180 DATA DE3761EBEBEBE030,1342
YG 10190 DATA D0EB20D160A5CD8D,1291
KR 10200 DATA 05D44CCE60E6CDA5,1195
MO 10210 DATA CDC908D01CA200B6,946
NC 10220 DATA CDBD3661186920 9D,863
GH 10230 DATA 36619003FE3761EB,936
IF 10240 DATA EBEBE030D0EB20D1,1420
AA 10250 DATA 60A5CD8D05D44C5F,995
YT 10260 DATA E4A97785CBA96185,1251
VX 10270 DATA CCA000AD376120EA,955
IJ 10280 DATA 60A002AD366120EA,848
WS 10290 DATA 6060484A4A4A4A20,592
OY 10300 DATA FA6068298FC820FA,988
HP 10310 DATA 6060C90A90061869,682
TL 10320 DATA 174C876118691091,493
BR 10330 DATA CB60BB1CBAA297F,1153
WJ 10340 DATA C960900A8A4C2761,795
TJ 10350 DATA C92090078A3BE920,843
QC 10360 DATA 4C27618A18694091,688
CZ 10370 DATA CBC000D0DD607070,1144
RZ 10380 DATA 70476C613000000,436
NS 10390 DATA 0000000000000000,0
ML 10400 DATA 0000000000000000,0
MP 10410 DATA 0000000000000000,0
MT 10420 DATA 0000000000000000,0
MX 10430 DATA 0000000000000000,0
TU 10440 DATA 000000000030427C,238
TL 10450 DATA 61412E612E04C4F43,559
KB 10460 DATA 4154494F4E3A2030,517
VM 10470 DATA 3030302055736520,589
JQ 10480 DATA 6A6F79737469636B,880
YX 10490 DATA 20746F206D6F7665,730
VD 10500 DATA 207665727469636 1,782
TQ 10510 DATA 6C6C792E,383
```

Listing 2.

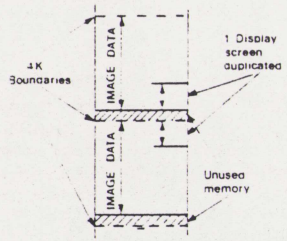


Figure 2.

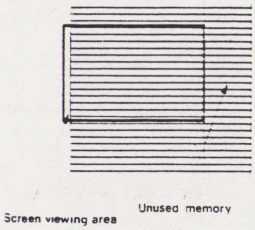


Figure 3.

```
0100 ;Demo of smooth horizontal scrolling.
0110 ;Hardware registers...
0120 DMACTL  =      $D400   ;DMA control.
0130 HSCROL  =      $D404   ;Horizontal scroll.
0140 ;Operating system shadows...
0150 SDMCTL  =      $022F   ;DMA control.
0160 SDLSTL  =      $0230   ;Display list pointer low.
0170 SDLSTH  =      $0231   ;Display list pointer high.
0180 ;Operating system vectors...
0190 SETVBV  =      $E45C   ;Set VBI vector.
0200 VBI2    =      $E45F   ;VBI stage two vector.
0210 ;Display list equates...
0220 JMP     =      $01     ;Jump.
0230 WVB     =      $40     ;Wait for vertical blank flag.
0240 LMS     =      $40     ;Load memory scan flag.
0250 HS      =      $10     ;Horizontal scroll flag.
0260 B4      =      $30     ;Blank 4 lines.
0270 B8      =      $70     ;Blank 8 lines.
0280 ;Page zero variables...
0290         *=     $CB
0300 PTERO   *=     *+2     ;Pointer.
0310 POS     *=     *+1     ;Position in string.
0320 COUNT   *=     *+1     ;Scroll counter.
0330         *=     $6000
0340         PLA
0350         LDA    #$00    ;Turn off DMA.
0360         STA    SDMCTL
0370         STA    DMACTL
0380         LDY    #20     ;Correct strings.
0390         LDA    #STARS&$FF
```

CRACKING THE CODE continued

```
0400          STA    PZERO
0410          LDA    #STARS/256
0420          STA    PZERO+1
0430          JSR    CHCONV
0440          LDY    #MESSLEN
0450          LDA    #MESSAGE&$FF
0460          STA    PZERO
0470          LDA    #MESSAGE/256
0480          STA    PZERO+1
0490          JSR    CHCONV
0500          LDY    #43      ;Copy start of message to end.
0510 CPYMESS  LDA    MESSAGE,Y
0520          STA    COPY,Y
0530          DEY
0540          BPL    CPYMESS
0550          LDA    #DLIST&$FF ;New display list.
0560          STA    SDLSTL
0570          LDA    #DLIST/256
0580          STA    SDLSTH
0590          LDA    #0
0600          STA    COUNT    ;Zero fine scroll counter.
0610          STA    POS      ;Zero position in string.
0620          STA    HSCROL   ;Reset hardware scroll.
0630          LDA    #$22     ;DMA on with standard screen.
0640          STA    SDMCTL
0650          LDY    #VBI&$FF ;Install VBI.
0660          LDX    #VBI/256
0670          LDA    #6
0680          JSR    SETVBV
0690          RTS
0700 ;Vertical blank interrupt.
0710 VBI      DEC    COUNT    ;Decrement scroll count.
0720          BPL    FINE     ;If positive just fine scroll.
0730          LDA    #3       ;Else reset to max. value.
0740          STA    COUNT
0750          INC    POS      ;Increment position.
0760          LDA    POS
0770          CMP    #MESSLEN ;At end?
0780          BNE    NEXT     ;No, go to next character.
0790          LDA    #MESSAGE&$FF ;Yes, go to start.
0800          STA    DATA
0810          LDA    #MESSAGE/256
0820          STA    DATA+1
0830          LDA    #0
0840          STA    POS
0850          JMP    FINE     ;Update fine scroll.
0860 NEXT     INC    DATA     ;Go to next character.
0870          BNE    FINE
0880          INC    DATA+1
0890 FINE     LDA    COUNT    ;Update fine scroll.
0900          STA    HSCROL
0910 VBEXIT   JMP    VBI2     ;Back to O.S.
0920 ;Convert an ATASCII string to display characters.
0930 CHCONV   DEY              ;Y holds number of characters.
0940          LDA    (PZERO),Y ;Get character.
0950          TAX              ;Save it.
0960          AND    #$7F     ;Turn off high bit (inverse).
0970          CMP    #$60
0980          BCC    CH1
0990          TXA              ;Above $60 - no adjustment.
1000          JMP    CHOK
1010 CH1      CMP    #$20
1020          BCC    CH2
1030          TXA              ;Between $20 and $5F.
1040          SEC              ;Subtract $20.
1050          SBC    #$20
1060          JMP    CHOK
1070 CH2      TXA              ;Between $00 and $1F.
1080          CLC              ;Add $40.
1090          ADC    #$40
1100 CHOK     STA    (PZERO),Y ;save character back.
1110          CPY    #$00     ;Last character?
1120          BNE    CHCONV
1130          RTS              ;Return.
1140 ;Display list data.
1150 DLIST    .BYTE  B8,B8,B8,LMS+7
1160          .WORD  STARS    ;Top string.
1170          .BYTE  B4,LMS+HS+2
1180 DATA     .WORD  MESSAGE ;Scrolling message.
1190          .BYTE  B4,LMS+7
1200          .WORD  STARS    ;Bottom string.
1210          .BYTE  JMP+WVB
1220          .WORD  DLIST
1230 STARS    .BYTE  '********************'
1240 MESSAGE  .BYTE  '    This is a demo of smooth '
1250          .BYTE  'horizontal scrolling. This message '
1260          .BYTE  'goes on and on and on...'
1270 MESSLEN  =      *-MESSAGE ;Number of bytes in string.
1280 COPY     *=     *+44     ;Space for copy of string.
```

Listing 3.

```
QZ 10 DIM HEX$(16)
HX 20 LINE=10000:TRAP 100:J=0:START=24576
VA 30 READ HEX$,CHKSUM:SUM=0
AA 40 FOR I=1 TO 15 STEP 2
IG 50 D1=ASC(HEX$(I,I))-48:D2=ASC(HEX$(I+
   1,I+1))-48
KT 60 NUM=((D1-7*(D1>16))*16+(D2-7*(D2>16
   )))
LW 70 SUM=SUM+NUM:POKE START+J,NUM:J=J+1:
   NEXT I
LY 80 IF SUM=CHKSUM THEN LINE=LINE+10:GOT
   O 30
IN 90 ? 'Checksum error on this line:'
VO 95 LIST LINE:END
YS 100 PRINT 'Data in memory.'
RA 10000 DATA 68A9008D2F02BD00,604
HX 10010 DATA D4A014A9B685C3A9,1C49
RL 10020 DATA 3085CC20816A859,F3E
```

CRACKING THE CODE cont

```
FA 10030 DATA A9CA85CBA96085CC,1309
JW 10040 DATA 208160A02BB9CA60,943
IC 10050 DATA 9922618810F7A9A5,1017
IF 10060 DATA 8D3002A9608D3102,648
LB 10070 DATA A90085CEB5CD8D04,991
BJ 10080 DATA D4A922BD2F02A050,845
VY 10090 DATA A260A906205CE460,881
GZ 10100 DATA C6CE1025A90385CE,968
GL 10110 DATA E6CDA5CDC958D011,1319
EO 10120 DATA A9CA8DAD60A9608D,1187
IG 10130 DATA AE60A90085CD4C79,974
JX 10140 DATA 60EEAD60D003EEAE,1226
IT 10150 DATA 60A5CEBD04D44C5F,995
OL 10160 DATA E488B1CBAA297FC9,1283
NG 10170 DATA 6090048A4C9E60C9,913
YV 10180 DATA 2090078A38E9204C,718
LW 10190 DATA 9E608A18694091C3,933
IF 10200 DATA C000D0DD60707070,1053
DO 10210 DATA 47B6603052CA6030,825
BE 10220 DATA 47B66041A5602A2A,759
VO 10230 DATA 2A2A2A2A2A2A2A2A,336
VS 10240 DATA 2A2A2A2A2A2A2A2A,336
WZ 10250 DATA 2A2A202020205468,400
AF 10260 DATA 6973206973206120,633
FU 10270 DATA 64656D6F206F6620,698
YC 10280 DATA 736D6F6F74682068,802
AL 10290 DATA 6F72697A6F6E7461,886
WD 10300 DATA 6C207363726F6C6C,795
DG 10310 DATA 696E672E20546869,689
JE 10320 DATA 73206D6573736167,787
PD 10330 DATA 6520676F6573206F,706
DZ 10340 DATA 6E20616E64206F6E,702
GG 10350 DATA 20616E64206F6E2E,638
BP 10360 DATA 2E2E,92
```

Listing 4.

GAMES REVIEW

by Fred Meijer

Mega-review Richard Gore

Time for something new in the Twaug Newsletter, the Mega-review. In these types of reviews I will review a number of programmes, that are somehow related to each other. Instead of the 'normal' reviews I won't discuss all the details.

This mega-review is about the new programs of a 'new' publisher, namely Richard Gore from the U.K. I have especially not used the word 'company' because his software is sold exclusively by DGS in England and by KE Soft in Germany.

The first package published by Richard Gore is a package consisting of two games, namely 'Jawbreaker' and 'Mousekattack'. Maybe you think: mmm I know these titles...

GAMES REVIEW cont.

Yes, you are right! Because these are two very very old games. I could not find a release-date of 'Mousekattack', but I found the release-date of 'Jawbreaker' as I have the original package myself on my shelves. 'Jawbreaker' is a game from... 1980!!! So it is almost fifteen years old! But o.k. our Atari is also very very old and it is still a nice computer.

What are 'Jawbreaker' and 'Mousekattack' all about? Both games are variations of the famous game 'Pacman'. I think that everybody knows 'Pacman' so I won't explain the game. In 'Jawbreaker' the pacman has been replaced by little teeth which have to eat all the candy in the playfield. The four ghosts have been replaced by smileys. In every corner you can find four powerpills. When you eat one of those pills, you can eat the ghosts. So as I said above, it is an exact copy of 'Pacman'.

In 'Mousekattack' you don't have to eat the dots, but you have to place tubes in the labyrinth. When the whole labyrinth is filled with tubes, you go on to the next level. The ghosts of 'Pacman' have been replaced by rats, when they touch you, you lose a life. You can keep the rats at a save distance when you eat the cat symbols. However, sometimes you find that there are super rats that can eat the cat symbols.

The games are quite nice. The large disadvantage is, that they are old. Games from 1980 were not made to the same quality as nowadays. I also believe that almost everyone has an illegal copy of these games in their collection.

Summary Jawbreaker/Mousekattack:

Publisher: Richard Gore
Sold at : DGS & KE Soft
Price : £6.95 or DM 24.80

Graphics : 5
Sound : 4
Playability : 6
Value for money: 2
Overall : 4

Conclusion: Should be made Public Domain.

Then the next game: 'Arena'. The name will probably sound familiar too, because the first levels of this game had been published by Page 6 as a bonus on one of their issue disks.

What is 'Arena' all about? You have to lead a number of power-pods to the exit of the screen. You do this by changing the playfield so that the pods can go to the exit by themselves. When a pod touches a 'moving part' it will die.

This is in short the aim of the game. In my opinion the game does not look very good. Only simple and undetailed signs have been used and I can not see what these are supposed to be. There is almost no sound, just some rare sound effects.

I did take a look at the directory. Mmm Runtime.Obj.

GAMES REVIEW continued

Aha, so this game is a compiled Turbo Basic game. It is a 'normal' basic game that has been compiled into machine language, so that it runs somewhat faster. This does not have to mean that Turbo-Basic games are bad, because Turbo-Basic has some extra capabilities above Atari Basic. Take a look at 'Whoops !',where you cannot see that it has been programmed in Turbo Basic. However the extra capabilities of Turbo Basic have not been used in 'Arena'. Probably this is the result of lack of knowledge of the programmer. The original version published by Page 6 runs in 'normal' Atari-Basic. So this game is probably compiled for speed and protection of the listing.

Summary Arena

Publisher: Richard Gore
Sold at : DGS & KE Soft
Price : £5 or DM 19.80

Graphics : 3
Sound : 3
Playability : 5
Value for money: 4
Overall : 4

Conclusion: Should be made Public Domain

The next game is 'Bubble zone'. In this game you have to save the Earth by shooting as many bubbles as possible in the playfield with your MK42 tank. Sometimes you find 'Warp tokens' and when you have collected enough of these, you can go to the next level. Some bubbles change into various bonusses. When you get a bonus you receive for example, extra energy, extra shield etc. etc.

The graphics of the game look reasonable. The characters are big and clear. The animation, however, is not very smooth. The bubbles are changing their size very jerkily and it's the same with the movement, especially the control of the tank, it is very bad. The control is always too late and when you change direction you always drive forward. Just like 'Arena' there's almost no sound.

Just like the other games, I am not very impressed about this one. This is not a game which can be sold commercially. As 'Arena', 'Bubble zone' is a compiled turbo basic game.

Summary Bubble Zone

Publisher: Richard Gore
Sold at : DGS & KE Soft
Price : £5 or DM 19.80

Graphics : 4
Sound : 2
Playability : 2
Value for money: 3
Overall : 3

Conclusion: Should be made Public Domain

And now the last game. You probably think not another bad game! Sadly I have to disappoint you. The last game 'Alien Blast' is not a bad game, no, when you see this game you will want to cry.

The story: Basic version of Space Invaders. You have to shoot all the aliens before they reach the bottom of the screen. There is only one thing different with the original version, you have to shoot the aliens with your lightgun. It sounds like a nice idea.

After some loading you hear (only on a 130XE) a digitized voice saying: Alien Blast! The music sounds very familiar. This is not so strange, as the music is known as 'Tico Tico', it is a PD listing, which had been published in various magazines. At the bottom of the screen is a scroller, but not a smooth one. You see some text going from right to left in a very jerky way.

The game itself is obvious a type-in listing from some magazine. Even the aircraft which has to shoot the aliens and would normally be controlled with the joystick is still on the screen. In the first level you get 10 bullets to shoot 8 aliens. This is far to few, because you always will miss the target a few times. Most of the famous lightgun games, like 'Operation Blood', offer a moving cursor on the screen, so you can see where you are aiming at. Sadly 'Alien blast' does not have this capability. In other lightgun games with no cursor, you still see the bullet and where it hit. 'Alien Blast' does not offer this capability. You can hardly see where your shot hit.

Together with John Maris I have taken a look at the directory. Mmm, no Runtime.Obj, but a very big Autorun.Sys and Autorun.Bas. Like the previous games this is another Turbo-Basic game, but the programmer has not even taken the time to compile it. This also became clear when I pressed Control-1 during play: everything stood still. After a simple press on the reset button we could examine the listing. According to John, who programs games under the name TML, the listing was a complete mess. It was not structured and the extra capabilities and commands of Turbo-Basic has completely been missed out. For example, you can load a binary file in turbo-basic with the command BGET. The programmer of 'Alien blast' probably does not know that, because he used a seperate machine-language routine to do exactly the same effect. Maybe, it would be a good idea to send the programmer a manual of Turbo-Basic?

I think 'Alien blast' is terrible. Richard Gore should be ashamed to sell such a game commercially. Combining a Basic-listing, which has probably been published by a magazine, with a Basic type-in listing of some music, this is not a way to earn some money! A terrible game like 'Drag' from KE Soft is still much better than 'Alien Blast'. What a garbage.

Summary Alien Blast:

Publisher: Richard Gore
Sold at : DGS & KE Soft
Price : £4.95 or DM 19.80

GAMES REVIEW continued

Graphics : 1
Sound : 2
Playability : 1
Value for money: 0
Overall : 1

Conslusion: Should be thrown into the garbage can.

Overall conclusion: I am not impressed about these games and their supposed commercial qualities. I know that there is almost no new commercial software published nowadays, but I think this junk only gives a bad name to the Atari XL/XE. I hope that KE Soft and DGS will stop selling these games as soon as possible.

About KE Soft. For over two years now, Kemal has been writing in his German magazine Zong, that KE Soft stands for quality. He must be convinced that new programmes must differ from the software which already exists for the 8-bit, before he wants to sell it. He calls this the famous 'KE-Filter'. Often enough other companies are blamed that they are selling so much Polish software and old Atari cartridges. But now I am asking myself how large is the hole of _his_ filter, when such junk is coming through it. He has made probably a very profitable arrangement with the publisher, because I see no other reason why he should sell these programmes. So here is a question for Kemal (in German): "Kemal, warum verkaufen Sie doch solche slechte Spiele als 'Arena', 'Bubble zone' und speziell 'Alien Blast'?"

Fred Meijer

GAMES REVIEW

by Mark Stinson

THE DARK CRYSTAL

Dark Crystal is an illustrated adventure based on the Jim Henson movie of the same name. The object is to locate and restore the missing shard of a magic crystal before the "Great Conjunction" of three sons. I haven't seen the film, so I had no prior knowledge of the story's plot or characters. My comments here are based solely on the inherent qualities of the game and not on its value as a souvenir.

The program occupies both sides of three disks. Disk 1, side A is the main interpreter; the other sides contain picture data for the dozens of colour illustrations. On-Line thoughtfully provides a back-up utility that lets you copy the picture disks, which are subject to lots of wear and tear. The interpreter is copy-protected, however.

Game play is similar to On-Line's popular Wizard and the Princess and Mission: Asteroid adventures. The parser is of the simple two-word, verb-nown variety; multiple commands or complex sentenses are not allowed. Each game location has its own Hi-Res colour illustration which must be pulled off the disk. You can "flip away" the picture temporarily to view a listing of your last several commands.

Provisions are made for the saving and loading of up to 15 different game positions. The program also lets you format an extra game-save disk during the course of play -- a lifesaver if you are in a tough spot with no formatted disk handy.

The illustrations for Dark Crystal are supposed to have been digitized from actual movie stills. Details and colouration are still rather crude -- certainly not photographic -- but the selection includes a number of dramatic perspective and shading effects you don't often see in games of this type.

I spent the best part of an evening mapping out Dark Crystal. I wandered through almost fourty different locations and exhausted three of the five picture-disk sides. Aside from a few unavoidable encounters with characters telling you what to look for, nothing happened. There were no threatening situations, no puzzles, mazes or unusual objects to pick up, just cute little creatures peeking out from behind trees, and a couple of dead ends. The handsomely printed owner's guide tells you most of what you need to know about your mission; very little is left to the imagination.

Dark Crystal seems to be more concerned with recreating the events and scenery of the movie than providing a fun game. Despite the fine packaging and professional engineering, I still prefer On-Line's previous Hi-Res adventures for the Atari.

STAR TREK

It is the 23rd century. You are at the helm of the United Federation Planets' cruiser Enterprise, travelling through space. You have just received orders to dock with one of the Federation's starbases to be briefed on an upcoming mission. As you approach the station, it becomes obvious that something is very wrong. A score of Klingon Katinga-class heavy cruisers has descended from warp space; the starbase is under heavy bombardment. You must make a decision that could spell intergalactic war on one hand, or the death of hundreds of innocent persons aboard the station on the other. Grimly you prepare yourself and your ship to do battle with the Klingon invaders.

If this situation sounds like an outline for the next Star Trek film, you may be half right. Sega's new Star Trek: Strategic Operations Simulator takes many familier elements from the Trek films and TV series and combines them into an arcade-style action game.

The Operations Simulator puts you in the shoes of a cadet trainee at the Federation Academy's training school. You are in command of the Enterprise and all of its defensive weaponary. Using either the joystick or the keyboard, you must lead the Enterprise into the thick of battle, using photon torpedoes and phasers to combat the Klingons and save your beleaguered starbase. You also have the ability to use warp speed, a feature that allows you to outdistance your Klingon adverseries, or to speed to the defence of an overwhelmed starbase. The display screen is devided in to three parts. The lower third represents the Enterprise's bridge viewer.

GAMES REVIEW continued

The upper right third is a bird's-eye scanner that shows your relationship with the Klingons, starbase and other nearby objects. The remaining third is a bar-graph display that indicates your remaining photon torpedoes, warp energy, and the power remaining in your deflector screens. This particular gauge is one to watch, because if your shields become depleted, a disrupter-torpedo from a Klingon cruiser will mash you and your ship to jelly.

To be a seasoned Atari space-gamer, Star Trek may sound a lot like Star Raiders (a classic worth aspiring to). Purists will shake their heads and say "The first is always the best," and in this case I must agree with them. Although I am a big fan of Sega's Star Trek arcade game, this home version does not live up to the promise of its big brother. Game play is gratifyingly fast as you blast Klingons, and dock with starbase, avoid asteroids and engage in a stellar dogfight with the killer space probe, (Nomad.) (Trek Fans will recall that Nomad originally menaced the Empire in "The Changeling" episode.) But the home simulator is much too easy to master. I was able to rack up well over 200.000 points the second time I picked up my joystick. There is no multiplayer mode and only one skill level.

This incarnation of Star Trek probably won't impress a hard-core of an Atari computer gamer. Its lack of challenge would soon relegate it to the "I'm very bored"pile of games. As for myself, it will certainly be a long time before my next voyage on Sega's starship Enterprise.

GAMES REVIEW

by Mark Fenwick

WHEEL OF FORTUNE

You've seen the show, played the show, dreamt of the wonderful prizes, now play the game of 'Wheel of Fortune' on your Atari. You won't win a car or £10.000 in cash, but you will enjoy it anyway!

Wheel of Fortune written by Chet Walters is another quality PD title from the TWAUG collection for 2-4 players. Wheel of Fortune must be loaded with basic and the disk must be left without write protect otherwise it will not load!

On loading the game you are asked if you require instructions, these instructions are very much in depth and include how to create your own puzzle files. It's a good idea to check out the instructions first time around or print out a hard copy for later use should you decide to make up your own puzzle files.

Once loaded, you're asked how many players and their names. The computer will decide at random who will go first. You're asked to insert a puzzle file disk into the drive, don't look around for another disk as there are plenty of puzzle files on the program disk. When you choose a puzzle file, the catagories on it will be locked when played, so the next time you come to boot up the game and select the same puzzle file you won't get any questions you've had before. This process continues until all the files in a puzzle file are locked and

Once a puzzle file is selected the game begins, you'll be told who is first up. The screen then changes to show the blank letters and their headings which must be guessed, like the Television program the files consist of, 'Title, Person, Thing etc' You start off with the choice of Spinning the wheel, pausing the game, or solving the puzzle, though you can only pause the game at the beginning of a new game. All actions are carried out via the joystick and commands are very easy to follow. The spinning of the wheel is very well done as different values as well as Lose and Bankrupt scroll along the screen. It's very much a game of chance as there are five speeds at which the wheel turns depending how long you hold the fire button down, so a bankcrupt can fly straight by or creep up, or you could just miss that £1000!

Once you've safely landed some cash however much it may be, it's time to take a guess at a letter. The lower portion of the screen changes to show the whole alphabet underneath which is a line of consenants and vowels. Move left or right then fire to select a letter, your score will accumulate depending how many of your chosen letters are in the puzzle. You keep going until you guess a letter wrong then the following player takes a turn. All letters used by players will be indicated on the A-Z grid so hopefully you shouldn't pick a letter twice, a screen prompt will tell you if you do, but you'll lose your go! This process is identical to the game show so if you've watched the show you'll pick it up in no time at all. At the start of each turn you can opt to spin the wheel, solve the puzzle or when you've some cash, buy a vowel.

The game consists of four rounds after which the one with the most cash can go for the car, sorry, play the bonus round, five letters and a vowel. Keep the keyboard handy as you'll need it to enter the puzzle when you come to solve it or get all the consenants on all of the rounds, every other command is done via the joystick. Each time you start round one over again you start with no cash, but a running total of your total winnings is displayed after the final round, this gives another goal to aim for even if you don't always get to the bonus round.

Overall Wheel of Fortune is the closest you'll get to meeting Nicky Campbell, it's addictive as well as very entertaining. With the added option of making up your own puzzle files (provided you have Atariwriter) then there's little chance of boredom when you exhaust the puzzle files on the game disk and there's plenty to go at too. The other thing I like is that when you come to load up again you're never confronted by the same puzzles, with the clever way they're locked in Dos. For the price of a PD disk it offers lasting entertainment for the whole family!

SCRABBLE

Trying to get the family involved playing games on the Atari is no easy task, some opt for shoot 'em ups while others find strategy and puzzle games their pleasure. So wouldn't it be nice to play a game that's both challenging and entertaining which involves the whole family ? What's the game I hear you ask... Scrabble of course!

Friday Fun Scrabble by Les Howarth available from the TWAUG

GAMES REVIEW continued

PD library has all the attributes of it's original board game version, but has those little extra's only found with your Atari.

Most board games when converted to a computer format seem to fail in certain areas making you wonder...why convert it at all. With Scrabble however, this isn't the case, for a start you don't have to sit scratching your head (or other parts of your anatomy) while adding up a tripple letter, tripple word scores! A press of a button is all it takes.

Scrabble is written in basic, so boot up doesn't require the old option key. Make sure you don't write protect the disk as the program includes a high score save feature. Once the game begins to load a title screen will appear followed by a chance to see instructions if desired. Should you opt for instructions, you'll be given a quick run down on the basics of the game plus the controls needed. The option for instructions is given as a question yes or no so you need not have to read them every time.

Once the game has loaded you're asked how many players are going to play, up to eight players can play, where you all share the player one joystick. Decide who will go first then proceed to enter the names of all the players, the first name entered goes first. Once you've entered all the players names an option for Panic game is given. This is a real tester to those members of the family who feel they should be members of Mensa! as with this option chosen players must think of a word and place it on the grid within one minute. It's a good idea to go for panic game as games are quicker and less time is spent listening to that immortal phrase 'hmmmm' for 15 minutes at a time.

The Scrabble board grid takes up the majority of the screen, all the coloured squares double word etc are coloured as of the board game. A small border surrounds the grid to give various information such as letters left, shuffle option, players letters and current high score. Play is simple, you must guide a black cursor via the joystick, picking up and dropping letters as well as executing other commands are carried out by pressing fire. So there's no cheating once a word is placed on the grid the following player must decide if the word is ok or not, so watch your spelling! If the word is ok then the score will be calculated and displayed at the bottom of the screen and the next player takes a turn. When playing the panic game you might find that there's not enough time to place your word in the given time, as the timer is about to run out a sound will indicate a few seconds to go. A part word on the screen will not count obviously if it doesn't make sense so the following player must click on to 'No' for word ok, this will clear it from the grid and the player gets his letters back. There's also a shuffle feature which counts as a turn where all or some of your letters can be changed by pressing the cursor on the shuffle square, however, you will not know your new letter until you next take a turn. As with the board game play continues until all the letters have been used, the winner will then have his/her name and score saved to disk for next time, you can then opt to play again with more or less players.

All in all Friday Fun Scrabble is a great conversion to computer format and plays exceptionally well. Although written in basic it is well coloured and the main grid is very clear as are the large character set used. All the things from the board game are here, plus more, and yes you get a 50 point bonus for a seven letter word. Being a PD title Friday Fun Scrabble offers good value for money and is bound to appeal to most if not everyone!

THANK YOU MARK FOR ALL THE HELP AND SUPPORT YOU HAVE GIVEN US OVER THE LAST TWO YEARS. ALL THE BEST TO YOU AND YOUR WIFE FOR 1995.

THE OL'HACKERS ATARI USER GROUP INC.

O.H.A.U.G. is an all 8-bit user group in the State of New York, they are producing a bi-monthly first class informative newsletter on disk.

The disk is double sided full of news, views articles and bonus games and/or utilities. The disk has its own printing utility which you can use to read the content of the disk on screen or make hard copies.

A large PD Library is also available.

TWAUG is contributing to the OHAUG newsletter and vice versa.

For more information on how to join why not contact the President of this first class club by writing to:

A. Pignato
O.H.A.U.G.
3376 Ocean Harbor Drive
Oceanside, N.Y.11572
U.S.A.

TWAUG
8-BIT

T.W.A.U.G.
P.O.BOX No.8
WALLSEND
TYNE & WEAR
NE28 6DQ

TWAUG NEWSLETTER

ANNOUNCEMENT

First of all, we would like to wish everyone a belated happy new year, and thank you all for your continued support over the last two years. We hope you will continue to give us the same support for a long time to come.

Recently, we were able to obtain a very good supply of good quality blank disks. From time to time I have received letters from some of our subscribers saying that they were finding it hard to buy blank disks at a reasonable price. We are now able to supply disks at the following prices.

10 disks £1.50, or 25 disks for £3.50. Prices include postage and packing. All disks will be formatted before posting to make sure that they are all good.

As we are now able to get disks cheaper, we have decided that we can now reduce the price of our PD library disks. From the release of this issue, PD disks will be priced as follows.

Double sided disks, £1.50 each, any five for £6, any ten for £10.

Single sided disks £1 each, or have two single sided titles put on to one double sided disk for £1.50. This will mean that you can buy any ten single sided disks for £6 or better still, any 20 single sided titles for £10. Or you can pick and mix, for example: 10 singles and 5 doubles for the price of 10 doubles.

As you will see in our latest PD library update, we now have over 420 titles in our library, and we are always on the lookout for new titles to add to it. Many of you have given us a lot of support over the last two years by buying from the library, and we hope that many more of you will now be able to give us even more support by buying disks over the next year.

Would you like to be a reviewer for TWAUG?

We are sorry to say that Mark Fenwick will no longer be doing game reviews for us. Mark is not giving up his 8-bit entirely, he is just storing it away as he feels that at some time in the future, he will be coming back to it but at the moment, he is moving on to another machine. We would like to give our thanks to Mark for all the time and support he has given us over the last two years, and we wish him all the best for the future.

This means that we are now looking for someone who would like to do a regular games review column for us. You will have a free hand as to what games you would like to review, but as we receive new software we would like to have it reviewed, we will pass it on to whoever has taken on the job.

If anyone is interested, then please get in touch with Dave Ewens either by writing to the TWAUG address, or by telephoning Dave on 0191/2710086, anytime except Friday evening as that is computer club night. I look forward to hearing from you.

CONTENT

Side A of this issue 13 disk has another good selection of programmes for you to enjoy.

The first program is a space invaders type game. I found, as I started shooting at the space craft, that it was very difficult to hit the target, the space craft sensed that it was being shot at and moved either backwards or forwards out of the way. After a while I found a way to hit the target regularly.

Number two on the menu is a program that appeals I am sure to the younger chilren. The program asks some personal questions first, for instance your name, the person's sex, Male or Female, the age of the person and finally it asks if you want a recorded message played. That recorded message is of course "Happy Birthday", a picture of a birthday cake with candles and a figure of a person is drawn on the screen.

The third program is a drawing program. You are prompted to enter some variation factors, the rotational angle and colour. Type in a number and it draws a graph which you can print out to keep.

The fourth program lets you personalise your disks with a message up to 36 characters long and this message is displayed on the screen while the DOS.SYS is loading. Just follow the prompts and have a formatted disk at the ready.

The next program on the menu is a two player game which is controlled with the numeric keys. You remove some matches from a pyramid, you must put your thinking cap on to beat your opponent. The computer tells you at the beginning of the game who the winner will be, sometimes it is the last one who draws a match and sometimes that person is the loser.

The next programe is some sort of one armed bandit kind of game. The control of the ball is via the joystick plugged into port one. The aim is to drop the ball into the correct section that is worth the amount you collected the ball from, it can be a nicle or a dime. If you drop it into the wrong section you lose a go.

And the last one on side A is an adventure game, the instructions are displayed when you start up this game.

On side B is a demo program of PRINTLAB which is available from Micro Discount for £6.95, it is in ARCed form.

There is a printer driver for the Canon BJ-200 Bubble-Jet to use with Atariwriter Plus, by Ralph Bradley, with Doc files. You must use MACRO1.M65 with MAC-65 editor, read the doc file first. This is also by Ralph Bradley.

SCROLL1.BAS is a demo program mentioned in the Basic Tutorial article on page 4.

SONGS.ARC is the song program of Andrew C.Thompson's Coding Caper article.

ATARI SUPPORT from RICHARD GORE

ARENA: The full 50 level version of the excellent puzzle game that was a demo bonus on Page 6's issue disk.
Price: £5 ($10) XL/XE Disk only.

BUBBLE ZONE: A fast blast em set on a grid being invaded by ever expanding bubbles. Collect the tokens and see how far you can advance.
Price: £5 ($10) XL/XE Disk only.

JAWBREAKER: Classic maze game based on Pac-Man, munch the sweets but don't forget that dental hygene! Two different mazes supplied which were originally separate games.
Price £4.50 ($10)

MOUSEKATTACK: Another classic maze game, but this one involves you plumbing all the levels of Rat Alley. Addictive arcade action with a simultaneous two player mode.
Price: £4.50 ($10)

Jawbreaker and Mousekattack are available as a double pack for only £6.50 ($12). They are available on disk, tape and Rambit turbo tape, please state which when ordering.

YORKY 256K plug in memory upgrade with manual and support disk. Very few units left. Hurry this could be your last chance!
Price: £50 ($90) +p&p (UK £2, EC £3.50, world £5)

OTHER COMMERCIAL SOFTWARE.

I also have limited stocks of older commercial software on a periodic basis. All titles are new and still shrink wrapped with original (English) documentations. At the moment I have the following titles available:

Disks at £4.75 ($10) each: SILENT SERVICE, HALLEY PATROL, TAXICAB HILL, GORF, WINTER CHALLENGE & SUMMER GAMES.

ROMS at £6.95 ($14) each: GATO

All software prices include p&p to UK, overseas please add £2 per order unless otherwise quoted. Payment by cash, cheque (payable to Richard Gore) or IMO. Prices in dollars are for USA customers and include shipping but payment must be made in US dollars cash, preferably using registered post!

Contact address:

RICHARD GORE, 79 SPROTBROUGH ROAD, SPROTBROUGH, DONCASTER, DN5 8BW, ENGLAND

Telephone: (0302) 784642

Plus coming soon..... new software from Germany, GTRACKER (a new sample sequencer), Golf Tour 94 (a new golf game), Super Print-LAB XE and possibly (in the UK) the ICD/OSS product line such as SpartaDOS X, R-Time 8, Action!, MAC 65 etc.

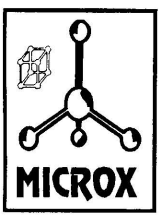
NB I will be at AMS 8, helping out on the DGS stand, watch out for some great offers. I hope to see you there...

Are You Still Complaining about lack of New software for your Atari 8 bit ?
If you are you don't know about Mail Order From

MICRO-DISCOUNT

265, CHESTER ROAD.
STREETLY.
WEST MIDLANDS.
B74 3EA.
ENGLAND.

TEL 021 353 5730
FAX 021 352 1669



ADAX *HANS KLOSS*

DARKNESS HOUR

The Atari Classic Programmer's Club

Would you like to become a life member with access to various programmer services, a helpline, regular printed newsletter and discounts off our software? If so, then we suggest you hurry. The life membership option is due to be abandoned after Christmas and all subsequent new memberships will be on a 12 months subscription basis only.

Current UK membership cost is:

12 months: £6.00
Life: £12.00

Overseas members most welcome. If you require more details then please send an SAE to the address below. Overseas Atarians, please send two International Reply Coupons (available from your post office) for more information.

Still Available:

Swift Spreadsheet (Standard) - £9.95

Swift Spreadsheet (New) - £12.95

(NOTE: New version includes a revised 40 page A4 manual. For details of the standard version and of the program itself, please refer to the review in issue 67 of New Atari User).

Available Soon

Menu Print: Still under development and currently undergoing its THIRD rewrite! We apologise for the delay but we expect the programming to be definitely completed by the end of September. More details will be published when ready.

THE ATARI CLASSIC PROGRAMMER'S CLUB
Pen-Tyddyn
Capel Coch
LLangefni
Anglesey
Gwynedd LL77 7UR
Wales

CURRENT NOTES

Helping Atari Owners Through the World of Computing.

Current Notes is published monthly (excluding January and August), in the U.S.

This magazine or you can call it newsletter has 80 pages full of computer news and very good articles that covers the Atari 8-bit and ST. It comes in full size of 11 inches by 8 1/2 inches.

SUBSCRIPTIONS:

Europe subscriptions is $69 per year.
Bankers drafts made payable to Current Notes
to: CN Subscriptions
122 N. Johnson RD.
Sterling, VA 20164

NOTE: VISA and MasterCard accepted.
Call (703) 450-4761

Editor's Note:
We at TWAUG are receiving the CN magazine monthly and we are always looking forward to it.

PHOENIX.

The new disk based news letter from Ireland, produced by Robert Paden.

PHOENIX a double sided disk, side 'A' will be packed full of text files containing Articles, reviews and much much more. Side 'B' will contain a good selection of PD software.

PHOENIX is only available from Robert Paden himself.